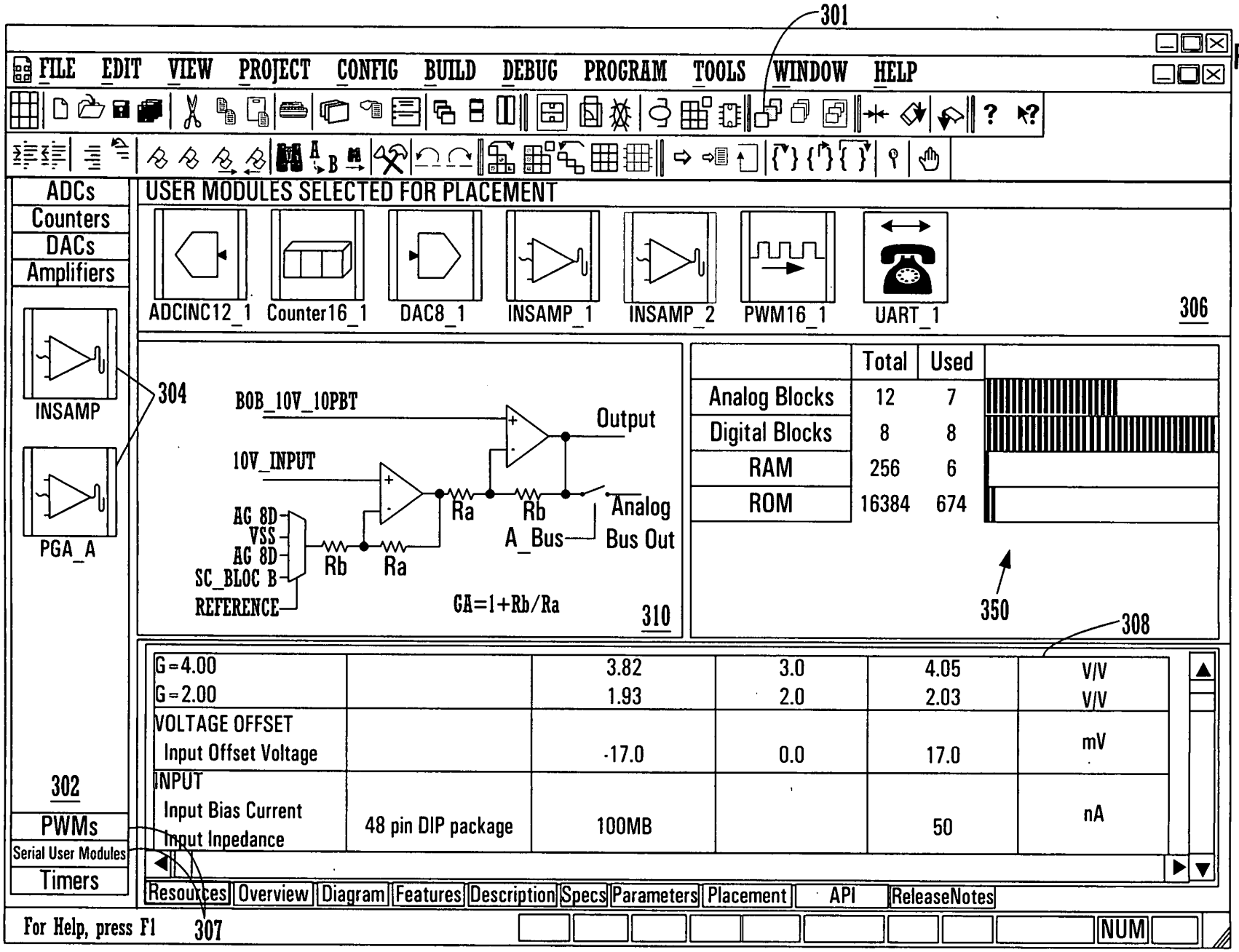
FIGURE 1A



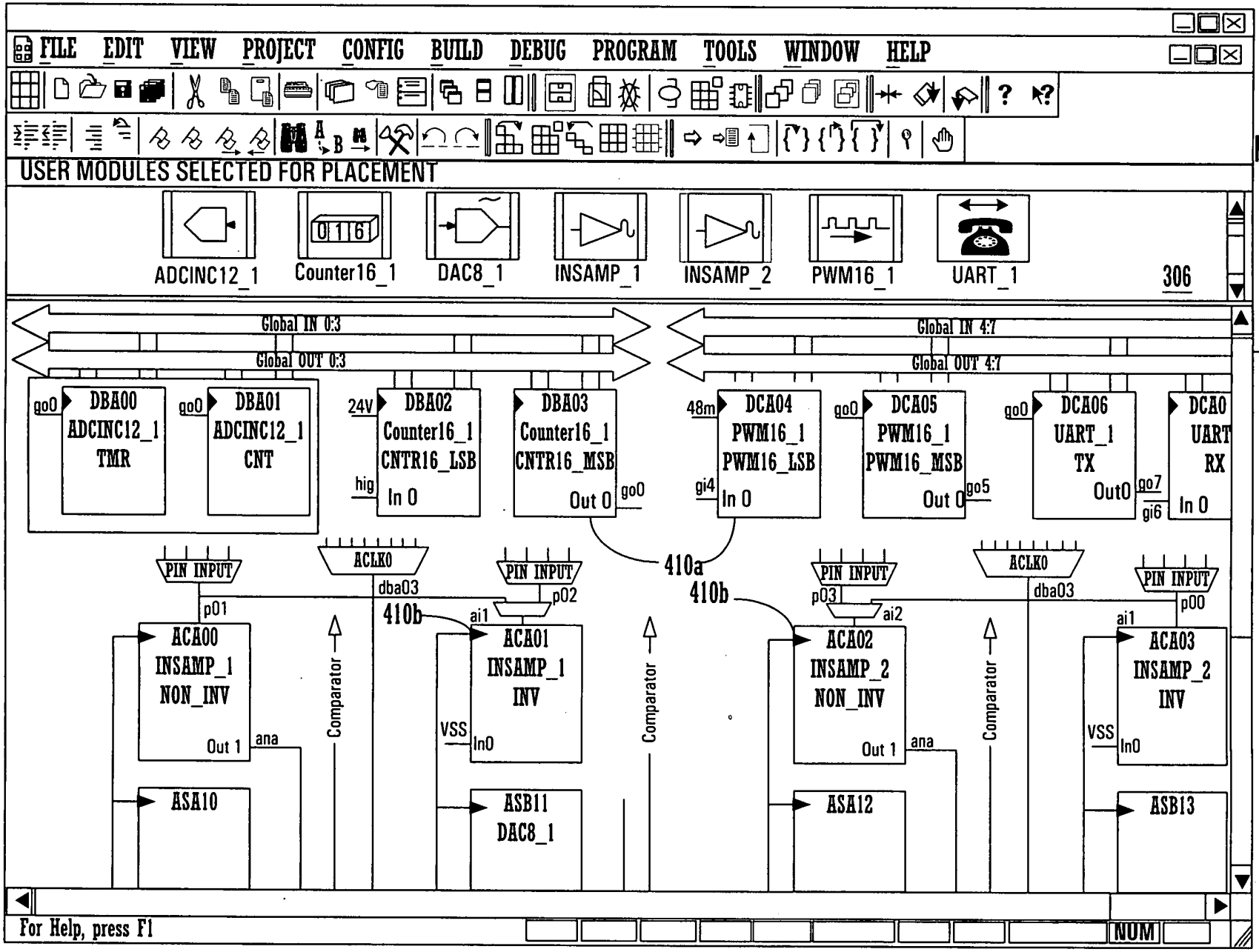

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
306
Global IN 0:3
Global OUT 0:3
Global IN 4:7
Global OUT 4:7
360
DBA00 ADCINC12_1 TMR
DBA01 ADCINC12_1 CNT
DBA02 Counter16_1 CNTR16_LSB
DBA03 Counter16_1 CNTR16_MSB
DCA04 PWM16_1 PWM16_LSB
DCA05 PWM16_1 PWM16_MSB
DCA06 UART_1 TX
DCA UART RX
410a
410b
410b
PIN INPUT
ACLK0
dba03
ACA00 INSAMP_1 NON_INV
ACA01 INSAMP_1 INV
ACA02 INSAMP_2 NON_INV
ACA03 INSAMP_2 INV
Comparator
ASA10
ASB11 DAC8_1
ASA12
ASB13
For Help, press F1
NUM

FIGURE 1B

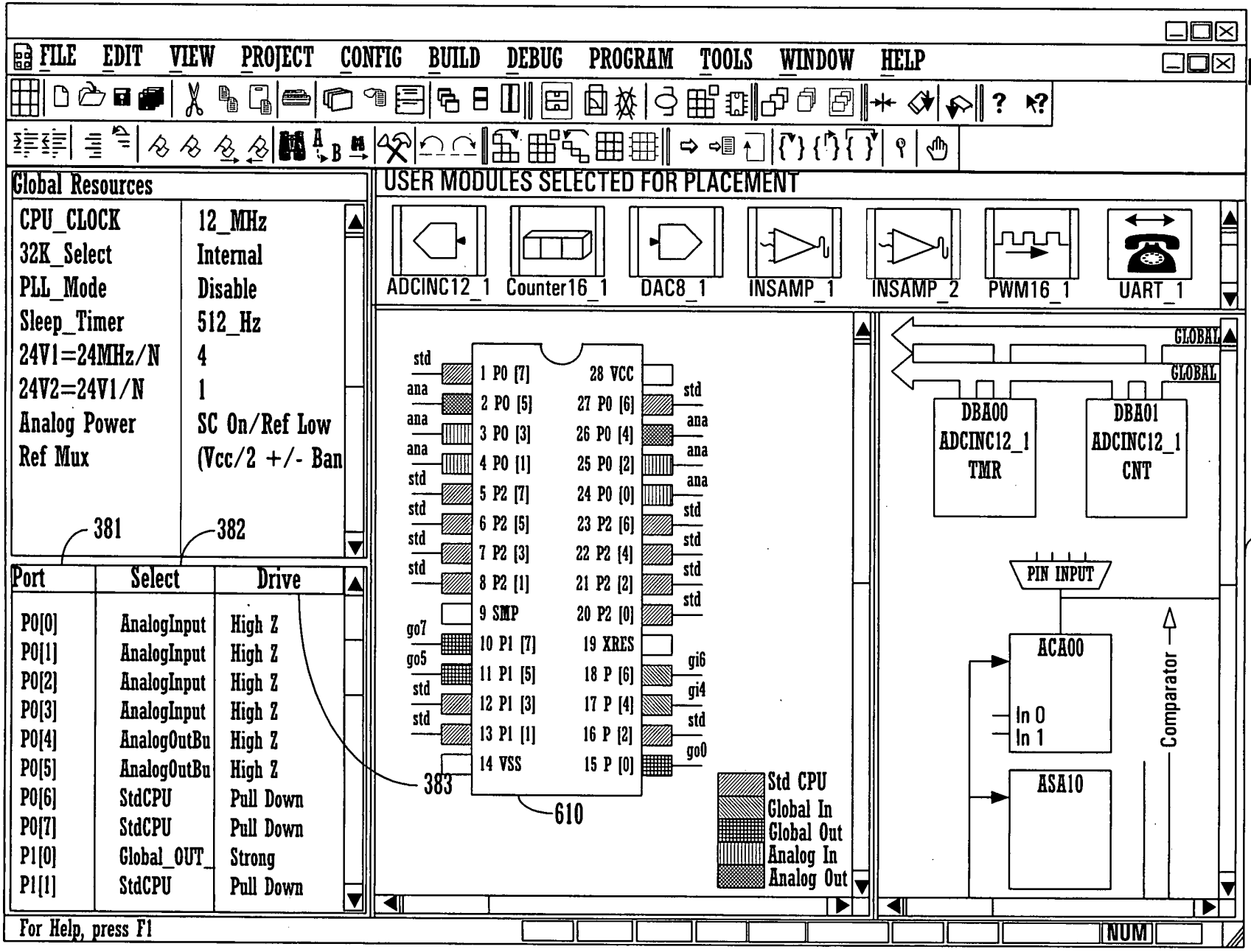

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global Resources
CPU_CLOCK 12_MHz
32K_Select Internal
PLL_Mode Disable
Sleep_Timer 512_Hz
24V1=24MHz/N 4
24V2=24V1/N 1
Analog Power SC On/Ref Low
Ref Mux (Vcc/2 +/- Ban
381
382
Port Select Drive
P0[0] AnalogInput High Z
P0[1] AnalogInput High Z
P0[2] AnalogInput High Z
P0[3] AnalogInput High Z
P0[4] AnalogOutBu High Z
P0[5] AnalogOutBu High Z
P0[6] StdCPU Pull Down
P0[7] StdCPU Pull Down
P1[0] Global_OUT_ Strong
P1[1] StdCPU Pull Down
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
1 P0 [7]
2 P0 [5]
3 P0 [3]
4 P0 [1]
5 P2 [7]
6 P2 [5]
7 P2 [3]
8 P2 [1]
9 SMP
10 P1 [7]
11 P1 [5]
12 P1 [3]
13 P1 [1]
14 VSS
28 VCC
27 P0 [6]
26 P0 [4]
25 P0 [2]
24 P0 [0]
23 P2 [6]
22 P2 [4]
21 P2 [2]
20 P2 [0]
19 XRES
18 P [6]
17 P [4]
16 P [2]
15 P [0]
383
610
Std CPU
Global In
Global Out
Analog In
Analog Out
GLOBAL
GLOBAL
DBA00 ADCINC12_1 TMR
DBA01 ADCINC12_1 CNT
PIN INPUT
ACA00
In 0
In 1
ASA10
Comparator
380
For Help, press F1
NUM

FIGURE 1C

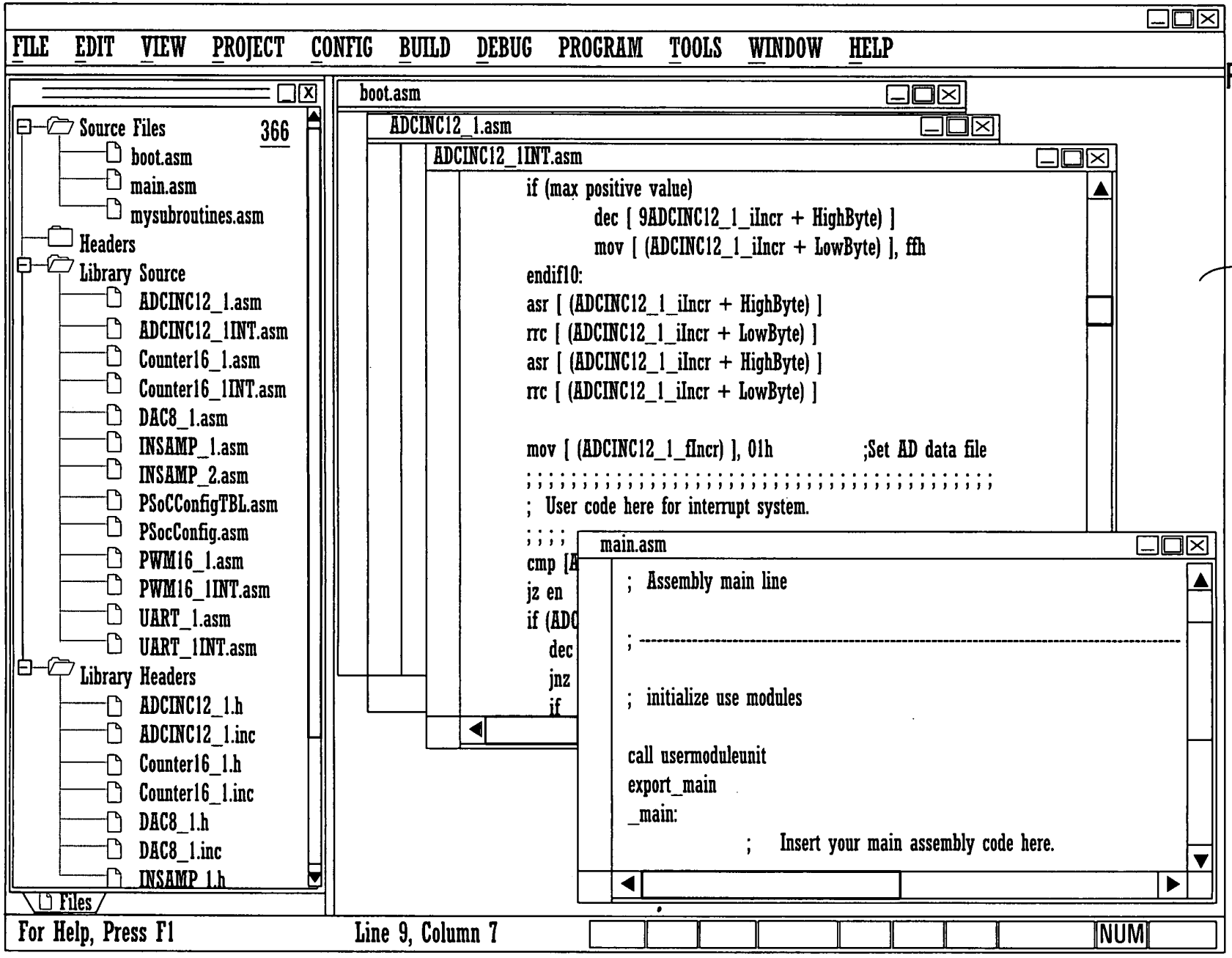


FIGURE 1D

365

4/34

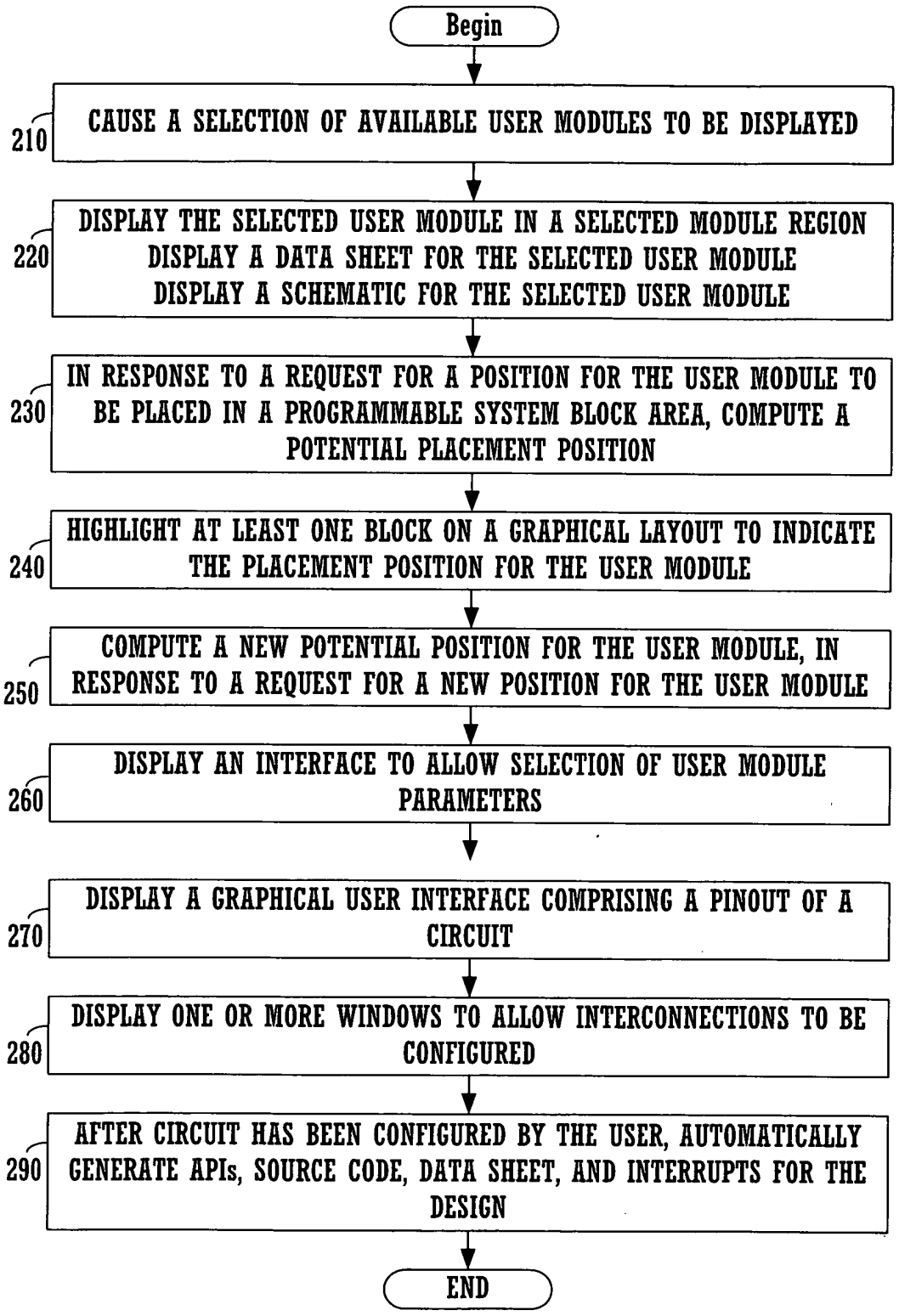

Begin
210
CAUSE A SELECTION OF AVAILABLE USER MODULES TO BE DISPLAYED
220
DISPLAY THE SELECTED USER MODULE IN A SELECTED MODULE REGION
DISPLAY A DATA SHEET FOR THE SELECTED USER MODULE
DISPLAY A SCHEMATIC FOR THE SELECTED USER MODULE
230
IN RESPONSE TO A REQUEST FOR A POSITION FOR THE USER MODULE TO BE PLACED IN A PROGRAMMABLE SYSTEM BLOCK AREA, COMPUTE A POTENTIAL PLACEMENT POSITION
240
HIGHLIGHT AT LEAST ONE BLOCK ON A GRAPHICAL LAYOUT TO INDICATE THE PLACEMENT POSITION FOR THE USER MODULE
250
COMPUTE A NEW POTENTIAL POSITION FOR THE USER MODULE, IN RESPONSE TO A REQUEST FOR A NEW POSITION FOR THE USER MODULE
260
DISPLAY AN INTERFACE TO ALLOW SELECTION OF USER MODULE PARAMETERS
270
DISPLAY A GRAPHICAL USER INTERFACE COMPRISING A PINOUT OF A CIRCUIT
280
DISPLAY ONE OR MORE WINDOWS TO ALLOW INTERCONNECTIONS TO BE CONFIGURED
290
AFTER CIRCUIT HAS BEEN CONFIGURED BY THE USER, AUTOMATICALLY GENERATE APIs, SOURCE CODE, DATA SHEET, AND INTERRUPTS FOR THE DESIGN
END

FIGURE 2

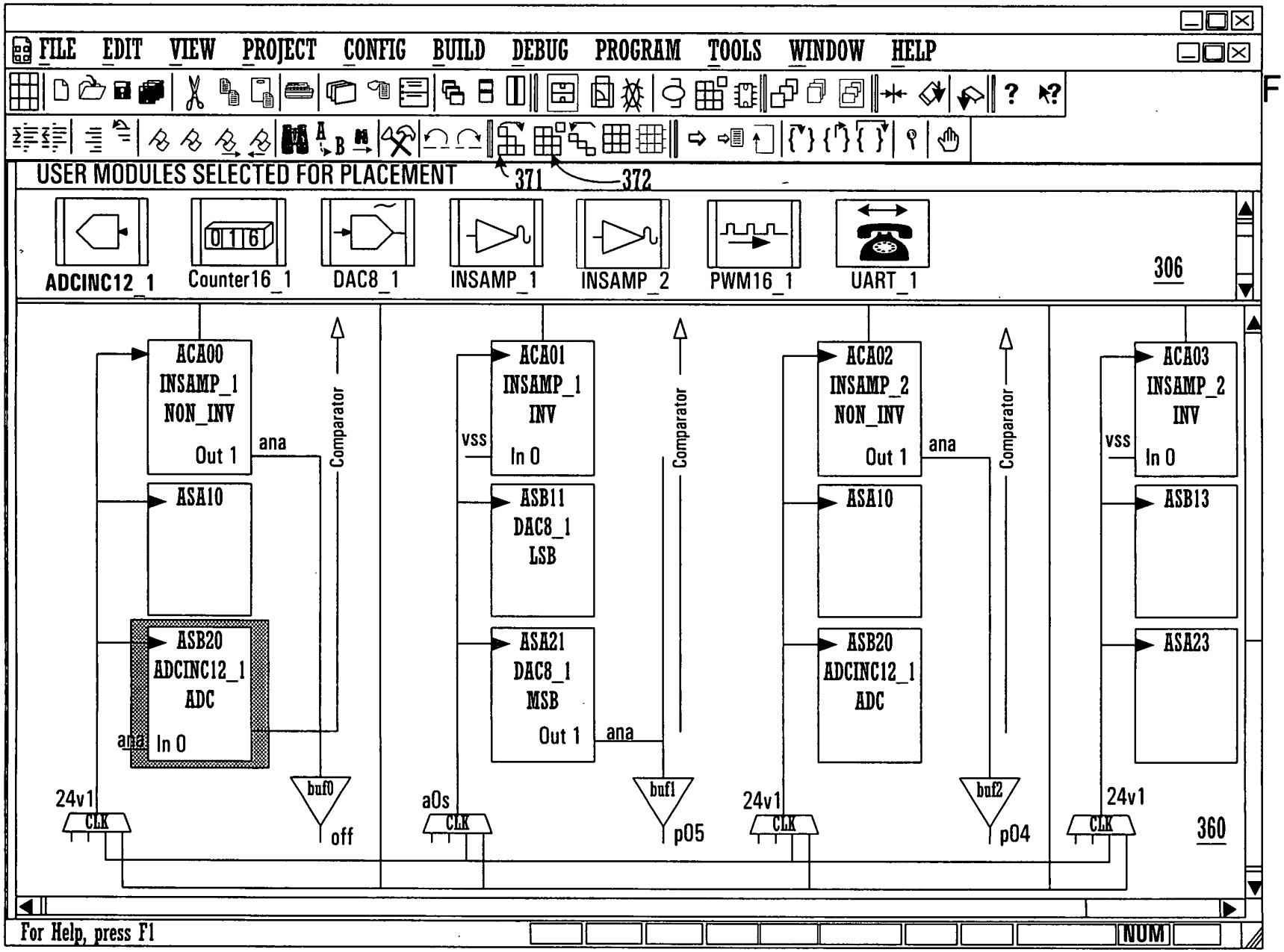

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
USER MODULES SELECTED FOR PLACEMENT
371
372
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
306
ACA00 INSAMP_1 NON_INV Out 1
ana
Comparator
ASA10
ASB20 ADCINC12_1 ADC ana In 0
buf0 off
24v1 CLK
ACA01 INSAMP_1 INV vss In 0
ASB11 DAC8_1 LSB
ASA21 DAC8_1 MSB Out 1
ana
Comparator
buf1 p05
a0s CLK
ACA02 INSAMP_2 NON_INV Out 1
ana
Comparator
ASA10
ASB20 ADCINC12_1 ADC
buf2 p04
24v1 CLK
ACA03 INSAMP_2 INV vss In 0
ASB13
ASA23
24v1 CLK
360
For Help, press F1
NUM

FIGURE 3A

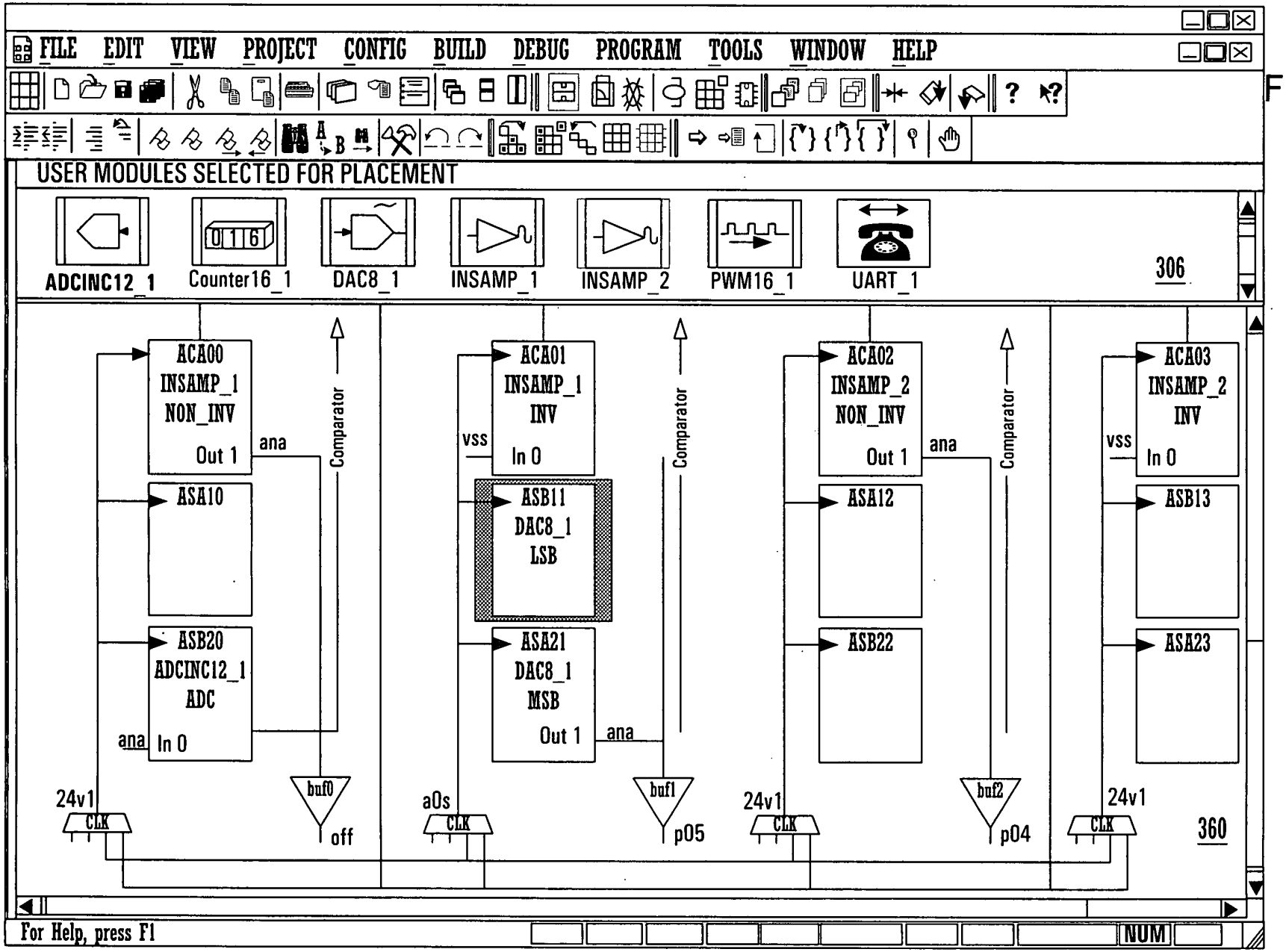

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
306
ACA00 INSAMP_1 NON_INV Out 1
ana
Comparator
ASA10
ASB20 ADCINC12_1 ADC ana In 0
buf0 off
24v1 CLK
ACA01 INSAMP_1 INV vss In 0
ASB11 DAC8_1 LSB
ASA21 DAC8_1 MSB Out 1 ana
buf1 p05
a0s CLK
ACA02 INSAMP_2 NON_INV Out 1
ASA12
ASB22
buf2 p04
ACA03 INSAMP_2 INV vss In 0
ASB13
ASA23
360
For Help, press F1
NUM

FIGURE 3B

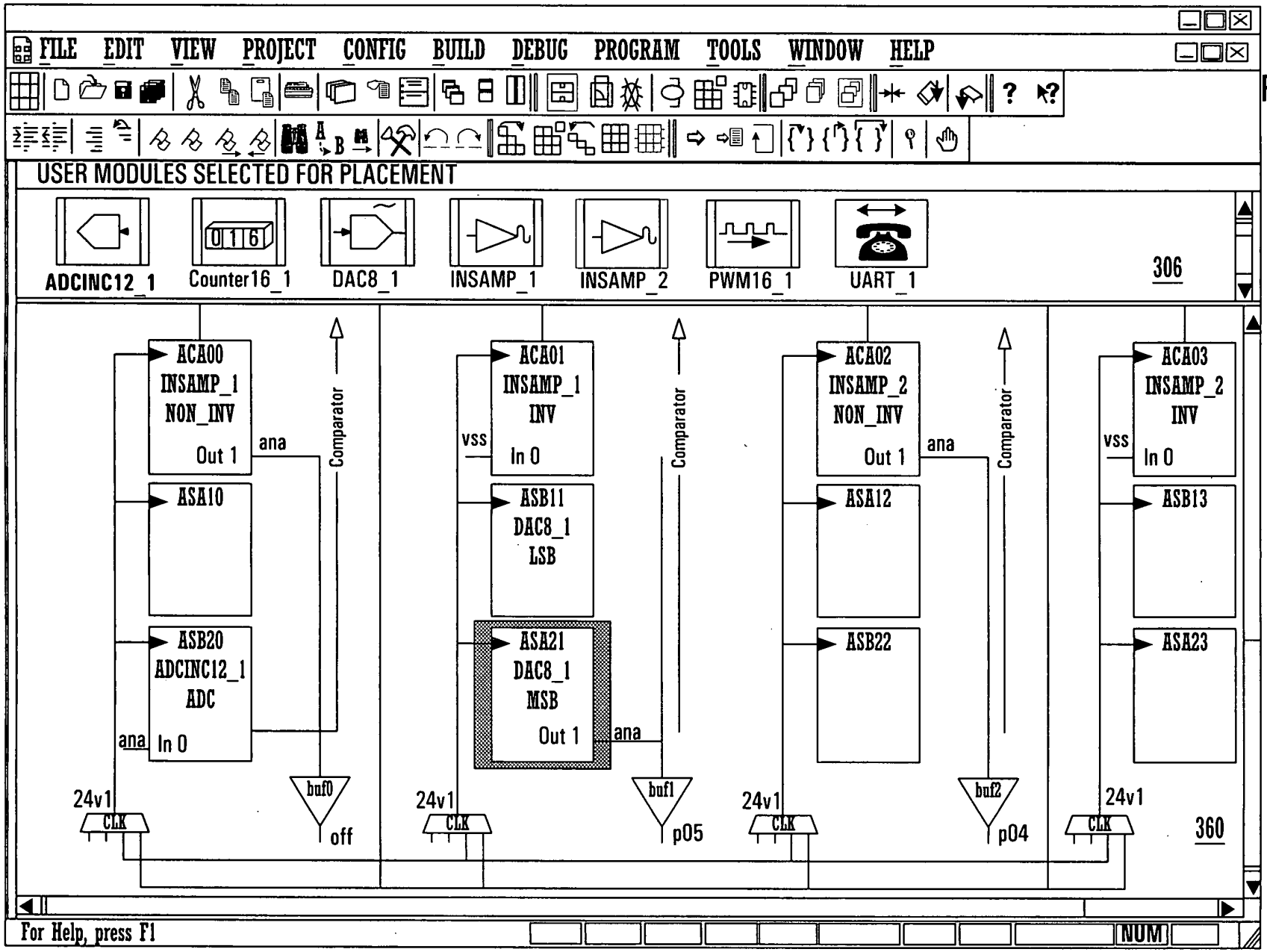

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
306
ACA00 INSAMP_1 NON_INV Out 1
ana
Comparator
ASA10
ASB20 ADCINC12_1 ADC ana In 0
buf0 off
24v1 CLK
ACA01 INSAMP_1 INV vss In 0
ASB11 DAC8_1 LSB
ASA21 DAC8_1 MSB Out 1
ana
buf1 p05
ACA02 INSAMP_2 NON_INV Out 1
ASA12
ASB22
buf2 p04
ACA03 INSAMP_2 INV vss In 0
ASB13
ASA23
360
For Help, press F1
NUM

FIGURE 3C

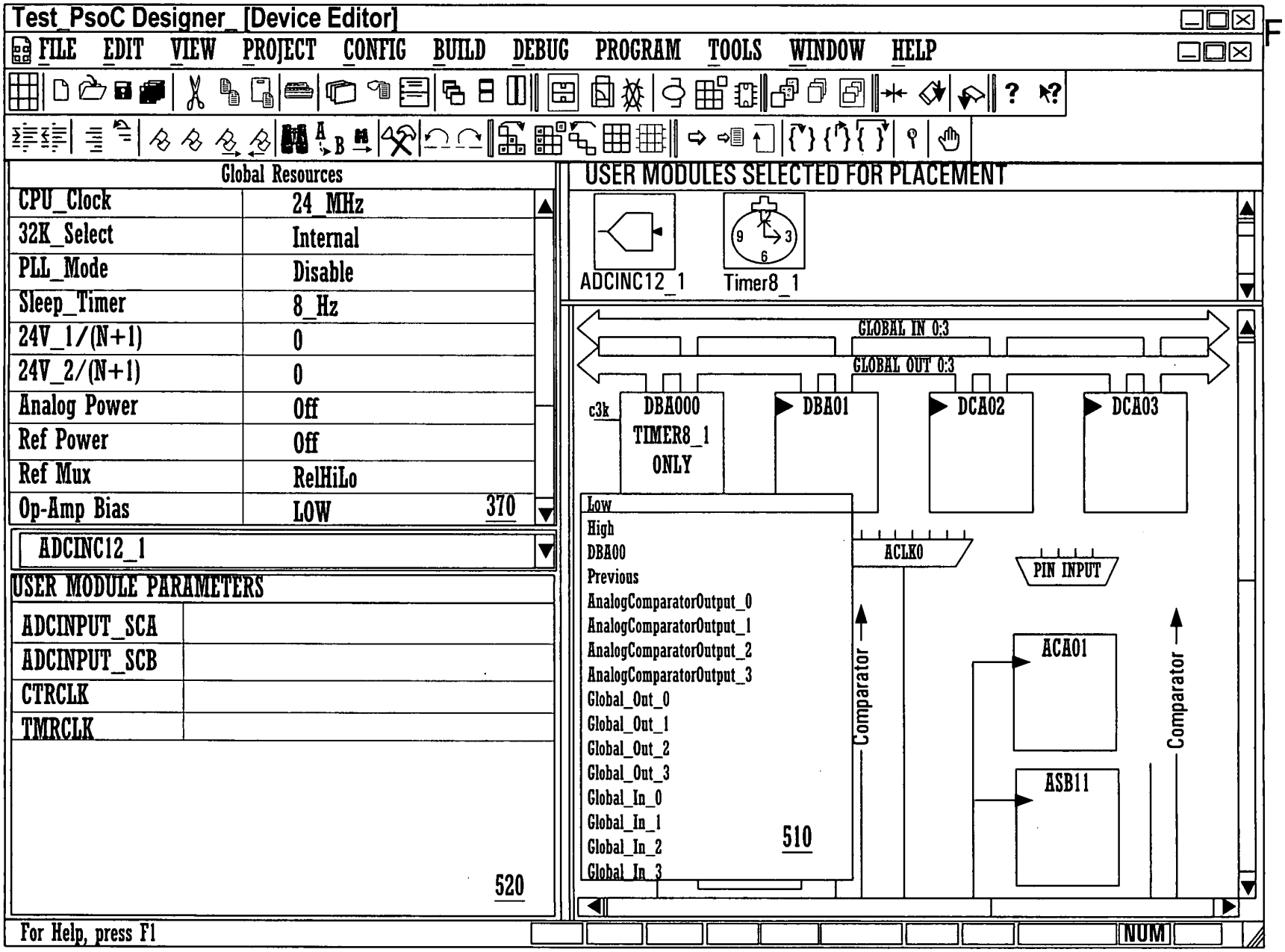

Test_PsoC Designer_ [Device Editor]
FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global Resources
CPU_Clock 24_MHz
32K_Select Internal
PLL_Mode Disable
Sleep_Timer 8_Hz
24V_1/(N+1) 0
24V_2/(N+1) 0
Analog Power Off
Ref Power Off
Ref Mux RelHiLo
Op-Amp Bias LOW
370
ADCINC12_1
USER MODULE PARAMETERS
ADCINPUT_SCA
ADCINPUT_SCB
CTRCLK
TMRCLK
520
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Timer8_1
GLOBAL IN 0:3
GLOBAL OUT 0:3
c3k
DBA000
TIMER8_1
ONLY
DBA01
DCA02
DCA03
Low
High
DBA00
Previous
AnalogComparatorOutput_0
AnalogComparatorOutput_1
AnalogComparatorOutput_2
AnalogComparatorOutput_3
Global_Out_0
Global_Out_1
Global_Out_2
Global_Out_3
Global_In_0
Global_In_1
Global_In_2
Global_In_3
510
ACLK0
PIN INPUT
Comparator
ACA01
ASB11
Comparator
For Help, press F1
NUM

FIGURE 4

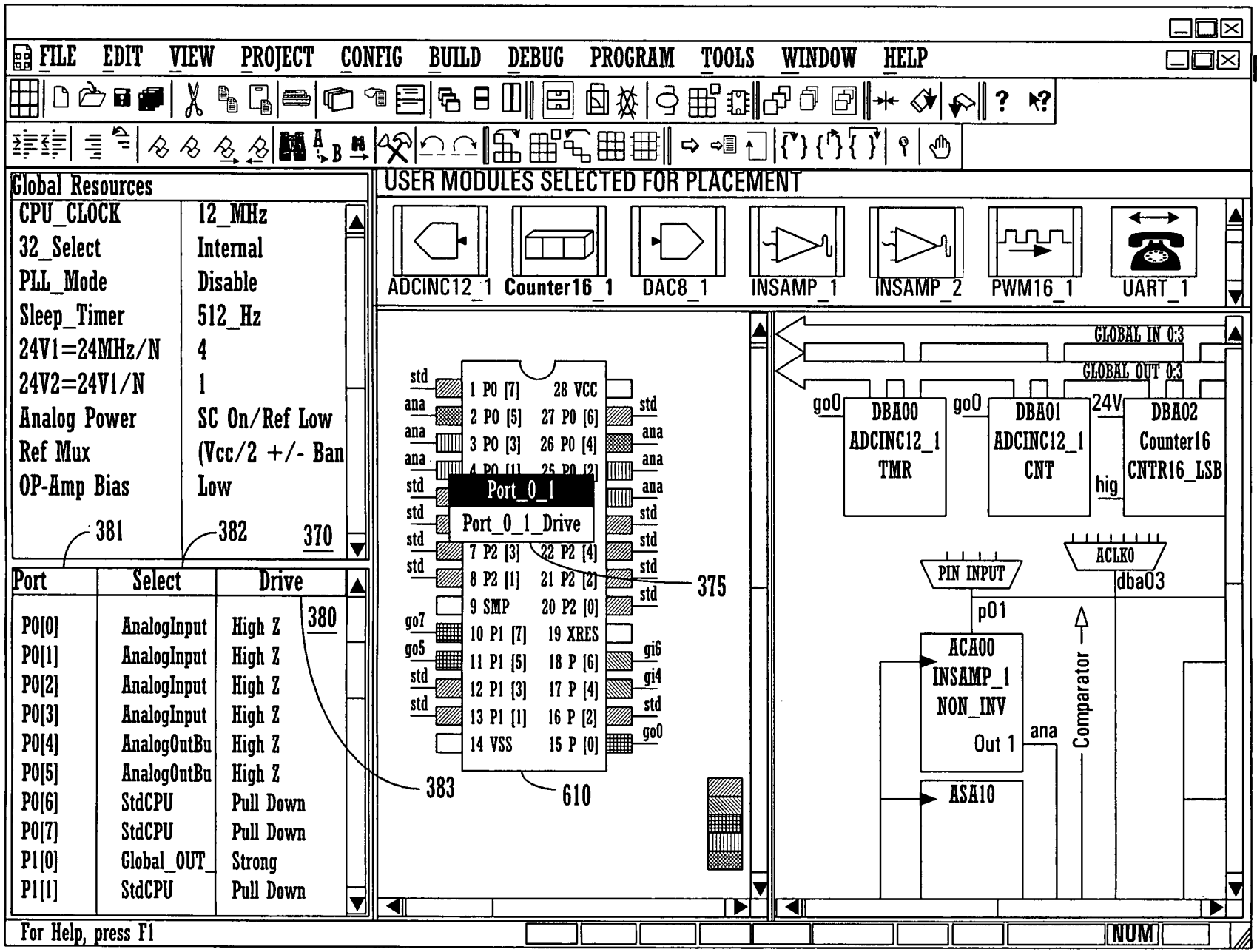

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global Resources
CPU_CLOCK 12_MHz
32_Select Internal
PLL_Mode Disable
Sleep_Timer 512_Hz
24V1=24MHz/N 4
24V2=24V1/N 1
Analog Power SC On/Ref Low
Ref Mux (Vcc/2 +/- Ban
OP-Amp Bias Low
381
382
370
Port Select Drive
P0[0] AnalogInput High Z
P0[1] AnalogInput High Z
P0[2] AnalogInput High Z
P0[3] AnalogInput High Z
P0[4] AnalogOutBu High Z
P0[5] AnalogOutBu High Z
P0[6] StdCPU Pull Down
P0[7] StdCPU Pull Down
P1[0] Global_OUT_ Strong
P1[1] StdCPU Pull Down
380
383
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
Port_0_1
Port_0_1_Drive
1 P0 [7]
2 P0 [5]
3 P0 [3]
7 P2 [3]
8 P2 [1]
9 SMP
10 P1 [7]
11 P1 [5]
12 P1 [3]
13 P1 [1]
14 VSS
28 VCC
27 P0 [6]
26 P0 [4]
22 P2 [4]
21 P2 [2]
20 P2 [0]
19 XRES
18 P [6]
17 P [4]
16 P [2]
15 P [0]
375
610
GLOBAL IN 0:3
GLOBAL OUT 0:3
DBA00 ADCINC12_1 TMR
DBA01 ADCINC12_1 CNT
DBA02 Counter16 CNTR16_LSB
24V
hig
ACLK0
dba03
PIN INPUT
p01
ACA00 INSAMP_1 NON_INV
Out 1
ana
Comparator
ASA10
For Help, press F1
NUM

FIGURE 5A

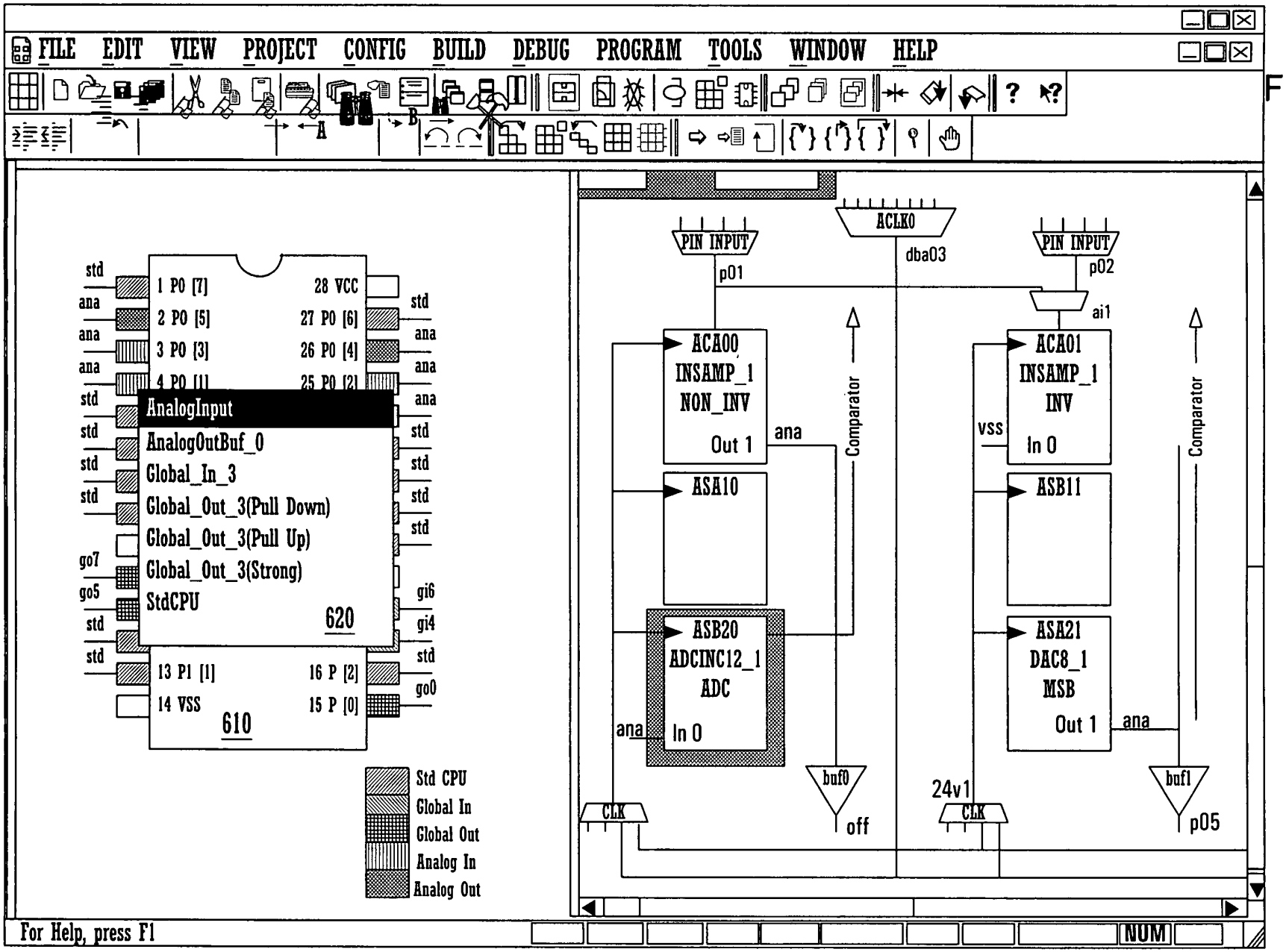

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
AnalogInput
AnalogOutBuf_0
Global_In_3
Global_Out_3(Pull Down)
Global_Out_3(Pull Up)
Global_Out_3(Strong)
StdCPU
620
610
1 P0 [7]
2 P0 [5]
3 P0 [3]
28 VCC
27 P0 [6]
26 P0 [4]
13 P1 [1]
14 VSS
16 P [2]
15 P [0]
Std CPU
Global In
Global Out
Analog In
Analog Out
ACLK0
dba03
PIN INPUT
p01
PIN INPUT
p02
ai1
ACA00
INSAMP_1
NON_INV
Out 1
ana
Comparator
ASA10
ASB20
ADCINC12_1
ADC
In 0
ana
buf0
off
CLK
ACA01
INSAMP_1
INV
vss
In 0
ASB11
ASA21
DAC8_1
MSB
Out 1
ana
Comparator
24v1
CLK
buf1
p05
For Help, press F1
NUM

FIGURE 5B

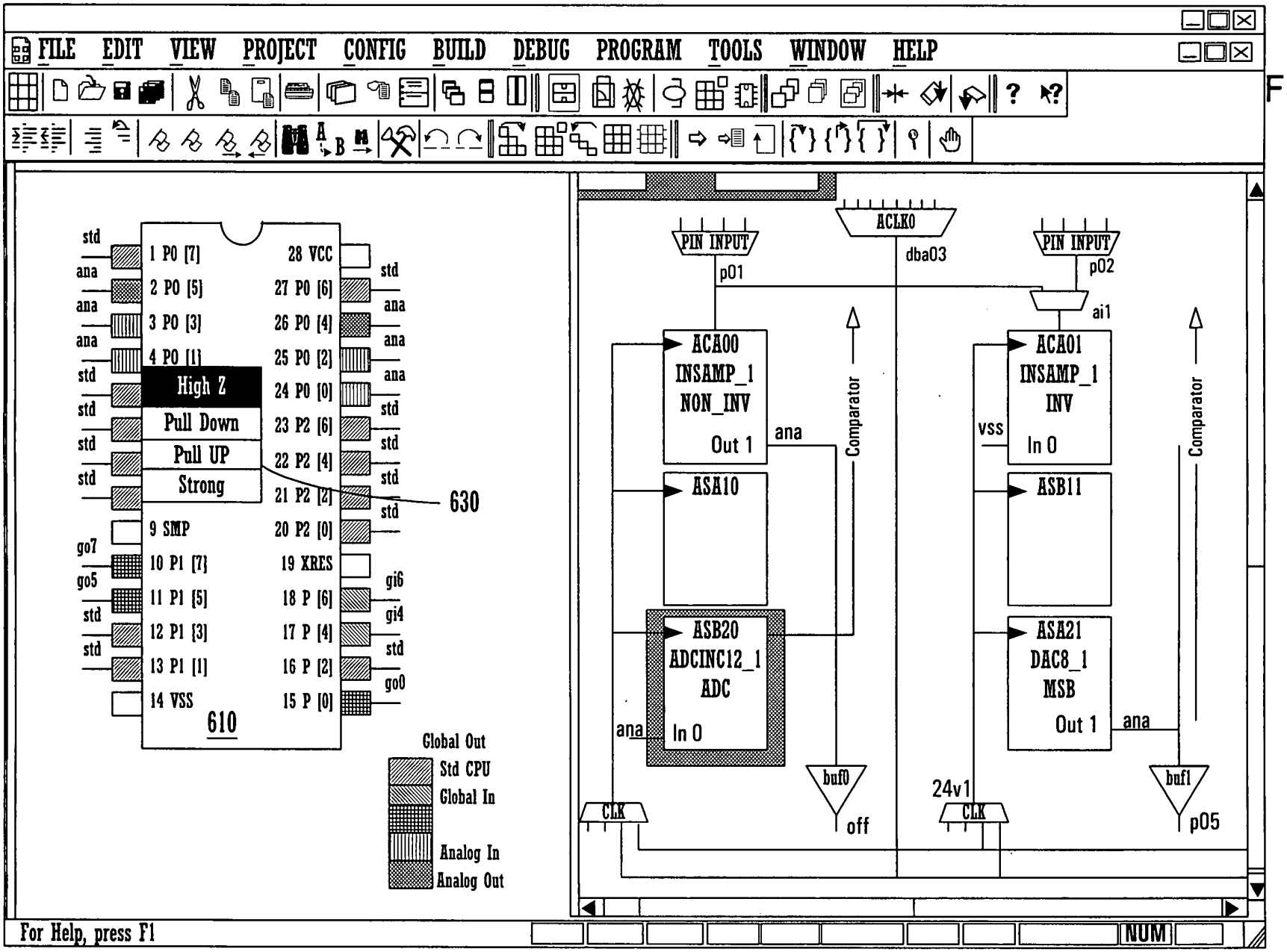

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
1 P0 [7]
2 P0 [5]
3 P0 [3]
4 P0 [1]
High Z
Pull Down
Pull UP
Strong
9 SMP
10 P1 [7]
11 P1 [5]
12 P1 [3]
13 P1 [1]
14 VSS
28 VCC
27 P0 [6]
26 P0 [4]
25 P0 [2]
24 P0 [0]
23 P2 [6]
22 P2 [4]
21 P2 [2]
20 P2 [0]
19 XRES
18 P [6]
17 P [4]
16 P [2]
15 P [0]
std
ana
go7
go5
gi6
gi4
go0
610
630
Global Out
Std CPU
Global In
Analog In
Analog Out
ACLK0
dba03
PIN INPUT
p01
PIN INPUT
p02
ai1
ACA00
INSAMP_1
NON_INV
Out 1
ana
Comparator
ASA10
ASB20
ADCINC12_1
ADC
In 0
ana
buf0
off
CLK
ACA01
INSAMP_1
INV
vss
In 0
ASB11
ASA21
DAC8_1
MSB
Out 1
ana
Comparator
24v1
CLK
buf1
p05
For Help, press F1
NUM

FIGURE 5C

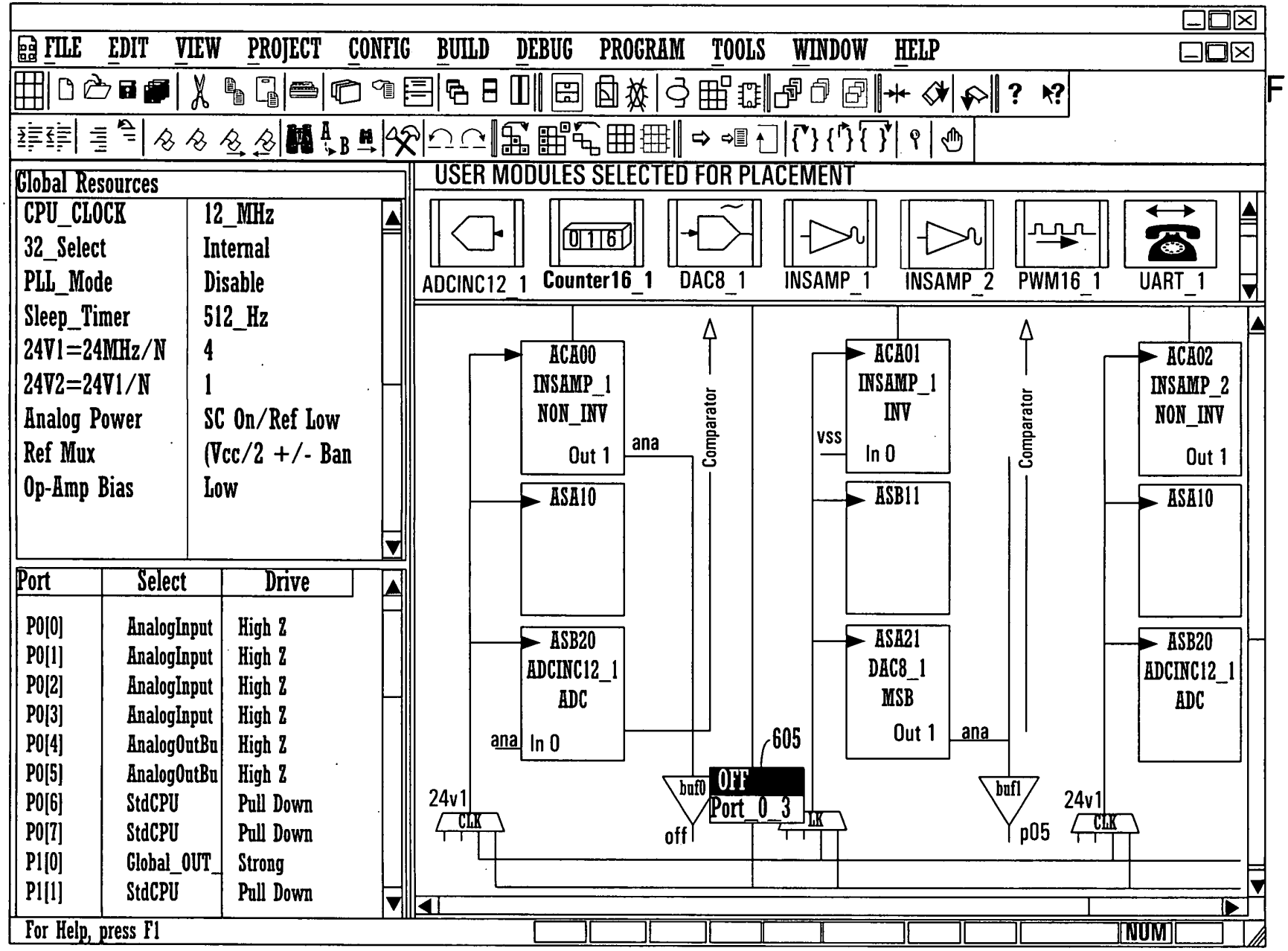

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global Resources
CPU_CLOCK 12_MHz
32_Select Internal
PLL_Mode Disable
Sleep_Timer 512_Hz
24V1=24MHz/N 4
24V2=24V1/N 1
Analog Power SC On/Ref Low
Ref Mux (Vcc/2 +/- Ban
Op-Amp Bias Low
Port Select Drive
P0[0] AnalogInput High Z
P0[1] AnalogInput High Z
P0[2] AnalogInput High Z
P0[3] AnalogInput High Z
P0[4] AnalogOutBu High Z
P0[5] AnalogOutBu High Z
P0[6] StdCPU Pull Down
P0[7] StdCPU Pull Down
P1[0] Global_OUT_ Strong
P1[1] StdCPU Pull Down
USER MODULES SELECTED FOR PLACEMENT
ADCINC12_1
Counter16_1
DAC8_1
INSAMP_1
INSAMP_2
PWM16_1
UART_1
ACA00 INSAMP_1 NON_INV Out 1
ana
Comparator
ASA10
ASB20 ADCINC12_1 ADC
ana In 0
24v1 CLK
ACA01 INSAMP_1 INV
vss In 0
ASB11
ASA21 DAC8_1 MSB Out 1
ana
Comparator
ACA02 INSAMP_2 NON_INV Out 1
ASA10
ASB20 ADCINC12_1 ADC
605
buf0 OFF Port_0_3
off
buf1
p05
24v1 CLK
For Help, press F1
NUM

FIGURE 6A

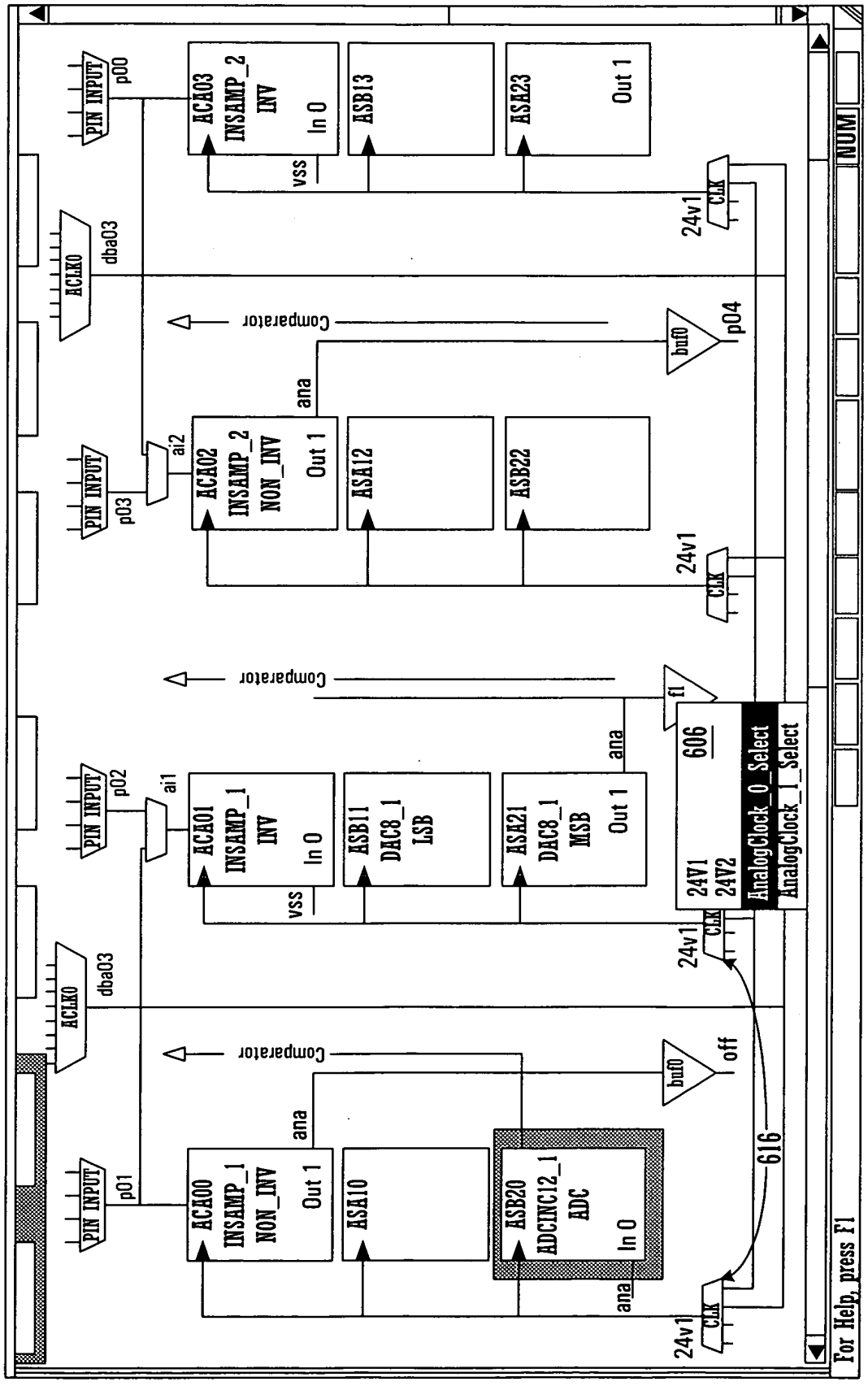

PIN INPUT
p00
ACA03
INSAMP_2
INV
In 0
vss
ASB13
ASA23
Out 1
24v1
CLK
ACLK0
dba03
Comparator
buf0
p04
ana
PIN INPUT
p03
ai2
ACA02
INSAMP_2
NON_INV
Out 1
ASA12
ASB22
24v1
CLK
Comparator
f1
PIN INPUT
p02
ai1
ACA01
INSAMP_1
INV
In 0
vss
ASB11
DAC8_1
LSB
ASA21
DAC8_1
MSB
Out 1
ana
606
24V1
24V2
AnalogClock_0_Select
AnalogClock_1_Select
24v1
CLK
ACLK0
dba03
Comparator
buf0
off
PIN INPUT
p01
ACA00
INSAMP_1
NON_INV
Out 1
ASA10
ASB20
ADCINC12_1
ADC
In 0
ana
616
24v1
CLK
NUM
For Help, press F1

FIGURE 6B

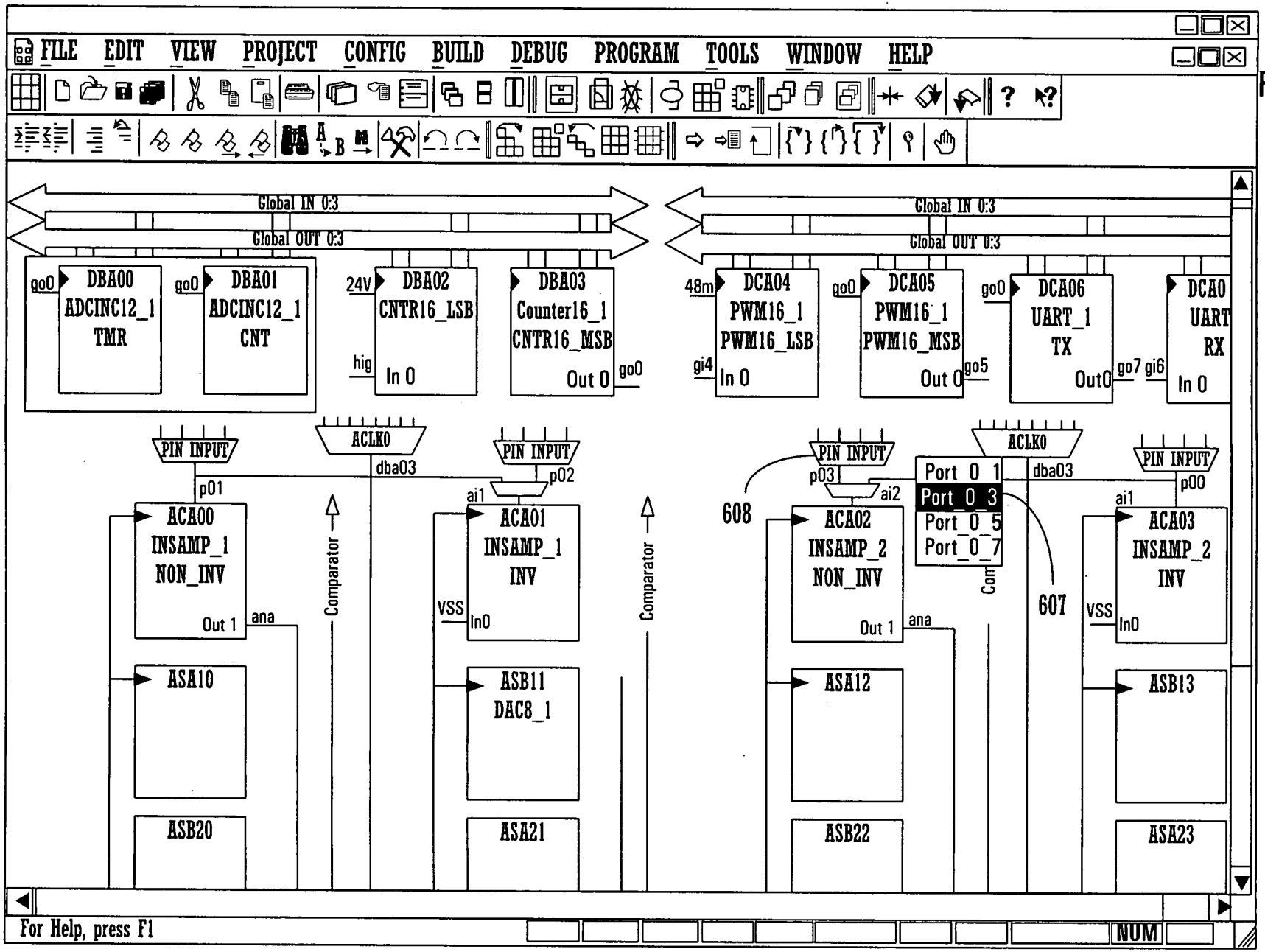

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global IN 0:3
Global OUT 0:3
go0
DBA00
ADCINC12_1
TMR
go0
DBA01
ADCINC12_1
CNT
24V
DBA02
CNTR16_LSB
hig
In 0
DBA03
Counter16_1
CNTR16_MSB
Out 0
go0
48m
DCA04
PWM16_1
PWM16_LSB
gi4
In 0
go0
DCA05
PWM16_1
PWM16_MSB
Out 0
go5
go0
DCA06
UART_1
TX
Out0
go7
gi6
DCA0
UART
RX
In 0
PIN INPUT
p01
ACLK0
dba03
PIN INPUT
p02
ai1
ACA00
INSAMP_1
NON_INV
Out 1
ana
Comparator
ACA01
INSAMP_1
INV
VSS
In0
Comparator
608
PIN INPUT
p03
ai2
ACA02
INSAMP_2
NON_INV
Out 1
ana
Port_0_1
Port_0_3
Port_0_5
Port_0_7
ACLK0
dba03
607
PIN INPUT
p00
ai1
ACA03
INSAMP_2
INV
VSS
In0
ASA10
ASB11
DAC8_1
ASA12
ASB13
ASB20
ASA21
ASB22
ASA23
For Help, press F1
NUM

FIGURE 6C

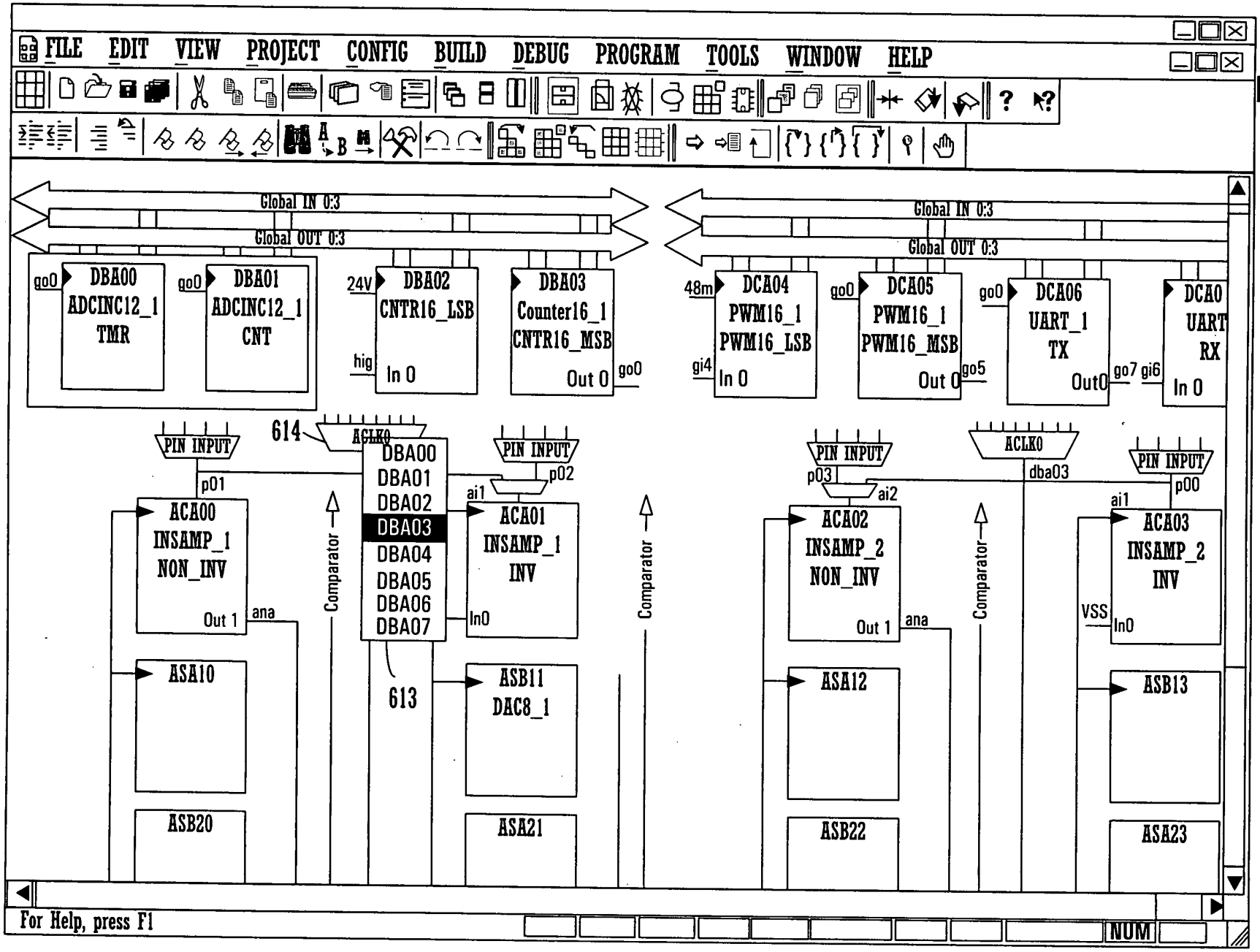

FILE EDIT VIEW PROJECT CONFIG BUILD DEBUG PROGRAM TOOLS WINDOW HELP
Global IN 0:3
Global OUT 0:3
go0
DBA00
ADCINC12_1
TMR
go0
DBA01
ADCINC12_1
CNT
24V
DBA02
CNTR16_LSB
hig
In 0
DBA03
Counter16_1
CNTR16_MSB
Out 0
go0
48m
DCA04
PWM16_1
PWM16_LSB
gi4
In 0
go0
DCA05
PWM16_1
PWM16_MSB
Out 0
go5
go0
DCA06
UART_1
TX
Out0
go7 gi6
DCA0
UART
RX
In 0
PIN INPUT
p01
ACA00
INSAMP_1
NON_INV
Out 1
ana
ASA10
ASB20
Comparator
614
ACLK0
DBA00
DBA01
DBA02
DBA03
DBA04
DBA05
DBA06
DBA07
613
PIN INPUT
p02
ai1
ACA01
INSAMP_1
INV
In0
ASB11
DAC8_1
ASA21
Comparator
PIN INPUT
p03
ai2
ACA02
INSAMP_2
NON_INV
Out 1
ana
ASA12
ASB22
Comparator
ACLK0
dba03
PIN INPUT
p00
ai1
ACA03
INSAMP_2
INV
VSS
In0
ASB13
ASA23
For Help, press F1
NUM

FIGURE 6D

```
configtbl.asm
;  Personalization tables
export LoadConfigTBL_project_Bank1
export LoadConfigTBL_project_Bank0
LoadConfigTBL_project_Bank1:
;  Global Register values
        db              61h, 03h        ;AnalogClockSelect register
        db              60h, 08h        ;AnalogColumnClockSelect register
        db              62h, 30h        ;Analog0Control register
        db              63h, 00h        ;AnalogModulatorControl register
        db              e1h, 30h        ;OscillatorControl_1 register
        db              00h, 00h        ;Port_0_DriveMode_0 register
        db              01h, 3fh        ;Port_0_DriveMode_1 register
        db              04h, a1h        ;Port_1_DriveMode_0 register
        db              05h, 50h        ;Port_1_DriveMode_1 register
        db              08h, 00h        ;Port_2_DriveMode_0 register
        db              09h, 00h        ;Port_2_DriveMode_1 register
        db              0ch, 00h        ;Port_3_DriveMode_0 register
        db              0dh, 00h        ;Port_3_DriveMode_1 register
        db              10h, 00h        ;Port_4_DriveMode_0 register
        db              11h, 00h        ;Port_4_DriveMode_1 register
        db              14h, 00h        ;Port_5_DriveMode_0 register
        db              15h, 00h        ;Port_5_DriveMode_1 register
        db              e3h, 84h        ;VoltageMonitorControl register
;  Instance name ADCINC12_1, User Module ADCINC12
;    Instance name ADCINC12_1, Block Name ADC(ASB20)
        db              90h, 90h        ;ADCINC12_1_AtoDcr0
        db              91h, 60h        ;ADCINC12_1_AtoDcr1
        db              92h, 60h        ;ADCINC12_1_AtoDcr2
        db              93h, f0h        ;ADCINC12_1_AtoDcr3
;    Instance Name ADCINC12_1, Block Name CNT(DBA01)
        db              24h, 21h        ;ADCINC12_1_CounterFN
        db              25h, 48h        ;ADCINC12_1_CounterSL
        db              26h, 00h        ;ADCINC12_1_Counter0S
;    Instance Name ADCINC12_1, Block Name TMR(DBA00)
        db              20h, 20h        ;ADCINC12_1_TimerFN
        db              21h, 18h        ;ADCINC12_1_TimerSL
        db              22h, 00h        ;ADCINC12_1_Timer0S
;  Instance name Counter16_1, User Module Counter 16
;     Instance name Counter16_1, Block Name CNTR16_LSB(DBA02)
        db              28h, 01h        ;Counter16_1_FUNC_LSB_REG
        db              29h, 16h        ;Counter16_1_INPUT_LSB_REG
        db              2ah, 00h        ;Counter16_1_OUTPUT_LSB_REG
;     Instance name Counter16_1, Block Name CNTR16_MSB(DBA03)
        db              2ch, 21h        ;Counter16_1_FUNC_MSB_REG
        db              2dh, 36h        ;Counter16_1_INPUT_MSB_REG
        db              2eh, 04h        ;Counter16_1_OUTPUT_MSB_REG
```

FIGURE 7A

```
; Instance name DAC8_1, User Module DAC8
;    Instance name DAC8_1, Block Name LSB(ASB11)
        db              84h, 80h        ;DAC8_1_LSB_CR0
        db              85h, 80h        ;DAC8_1_LSB_CR1
        db              86h, 20h        ;DAC8_1_LSB_CR2
        db              87h, 30h        ;DAC8_1_LSB_CR3
;    Instance name DAC8_1, Block Name MSB(ASA21)
        db              94h, a0h        ;DAC8_1_MSB_CR0
```

FIGURE 7A (Continued)

```
        db              95h, 41h        ;DAC8_1_MSB_CR1
        db              96h, a0h        ;DAC8_1_MSB_CR2
        db              97h, 30h        ;DAC8_1_MSB_CR3
; Instance name INSAMP_1, User Module INSAMP
;    Instance name INSAMP_1, Block Name INV(ACA01)
        db              75h, beh        ;INSAMP_1_INV_CR0
        db              76h, 21h        ;INSAMP_1_INV_CR1
        db              77h, 20h        ;INSAMP_1_INV_CR2
;    Instance name INSAMP_2, Block Name NON-INV(ACA00)
        db              71h, 3ch        ;INSAMP_1_NON_INV_CR0
        db              72h, a1h        ;INSAMP_1_NON_INV_CR1
        db              73h, 20h        ;INSAMP_1_NON_INV_CR2
; Instance name INSAMP_2, User Module INSAMP
;    Instance name INSAMP_2, Block Name INV(ACA03)
        db              7dh, ceh        ;INSAMP_2_INV_CR0
        db              7eh, 21h        ;INSAMP_2_INV_CR1
        db              7fh, 20h        ;INSAMP_2_INV_CR2
;    Instance name INSAMP_2, Block Name NON_INV(ACA02)
        db              79h, 2ch        ;INSAMP_2_NON_INV_CR0
        db              7ah, a1h        ;INSAMP_2_NON_INV_CR1
        db              7bh, 20h        ;INSAMP_2_NON_INV_CR2
; Instance name PWM16_1, User Module PWM16
;    Instance name PWM16_1, Block Name PWM16_LSB(DCA04)
        db              30h, 01h        ;PWM16_1_FUNC_LSB_REG
        db              31h, c4h        ;PWM16_1_INPUT_LSB_REG
        db              32h, 00h        ;PWM16_1_OUTPUT_LSB_REG
;    Instance name PWM16_1, Block Name PWM16_MSB(DCA05)
        db              34h, 21h        ;PWM16_1_FUNC_MSB_REG
        db              35h, 34h        ;PWM16_1_INPUT_MSB_REG
        db              36h, 05h        ;PWM16_1_OUTPUT_MSB_REG
; Instance name UART_1, User Module UART
;    Instance name UART_1, Block Name RX(DCA07)
        db              3ch, 05h        ;UART_1_RX_FUNC_REG
        db              3dh, e1h        ;UART_1_RX_INPUT_REG
        db              3eh, 00h        ;UART_1_RX_OUTPUT_REG
;    Instance name UART_1, Block Name TX(DCA06)
        db              38h, 0dh        ;UART_1_TX_FUNC_REG
        db              39h, 01h        ;UART_1_TX_INPUT_REG
        db              3ah, 07h        ;UART_1_TX_OUTPUT_REG
        db              ffh
```

FIGURE 7B

```
LoadConfigTBL_project_Bank0:
; Global Register values
        db              60h, 14h        ;AnalogColumnInputSelect register
        db              63h, 05h        ;AnalogReferenceControl register
        db              65h, 00h        ;AnalogSyncControl register
        db              e6h, 00h        ;DecimatorControl register
        db              02h, 00h        ;Port_0_Bypass register
        db              06h, f1h        ;Port_1_Bypass register
        db              0ah, 00h        ;Port_2_Bypass register
        db              0eh, 00h        ;Port_3_Bypass register
        db              12h, 00h        ;Port_4_Bypass register
        db              16h, 00h        ;Port_5_Bypass register
;  Instance name ADCINC12_1, User Module ADCINC12
;     Instance name ADCINC12_1, Block Name ADC(ASB20)
;     Instance name ADCINC12_1, Block Name CNT(DBA01)
```

FIGURE 7B (Continued)

```
        db              27h, 00h        ;ADCINC12_1_CounterCR0
        db              25h, 00h        ;ADCINC12_1_CounterDR1
        db              26h, 00h        ;ADCINC12_1_CounterDR2
;     Instance name ADCINC12_1, Block Name TMR(DBA00)
        db              23h, 00h        ;ADCINC12_1_TimerCR0
        db              21h, 00h        ;ADCINC12_1_TimerDR1
        db              22h, 00h        ;ADCINC12_1_TimerDR2
;  Instance name Counter16_1, User Module Counter16
;     Instance name Counter16, Block Name CNTR16_LSB(DBA02)
        db              2bh, 00h        ;Counter16_1_CONTROL_LSB_REG
        db              29h, 80h        ;Counter16_1_PERIOD_LSB_REG
        db              2ah, 64h        ;Counter16_1_COMPARE_LSB_REG
;     Instance name Counter16_1, Block Name CNTR16_MSB(DBA03)
        db              2fh, 00h        ;Counter16_1_CONTROL_MSB_REG
        db              2dh, 00h        ;Counter16_1_PERIOD_MSB_REG
        db              2eh, 00h        ;Counter16_1_COMPARE_MSB_REG
;  Instance name DAC8_1, User Module DAC8
;     Instance name DAC8_1, Block Name LSB(ASB11)
;     Instance name DAC8_1, Block Name MSB(ASA21)
;  Instance name INSAMP_1, User Module INSAMP
;     Instance name INSAMP_1, Block Name INV(ACA01)
;     Instance name INSAMP_1, Block Name NON_INV(ACA00)
;  Instance name INSAMP_2, User Module INSAMP
;     Instance name INSAMP_2, Block Name INV(ACA03)
;     Instance name INSAMP_2, Block Name NON_INV(ACA02)
;  Instance name PWM16_1, User Module PWM16
;     Instance name PWM16_1, Block Name PWM16_LSB(DCA04)
        db              33h, 00h        ;PWM16_1_CONTROL_LSB_REG
        db              31h, 37h        ;PWM16_1_PERIOD_LSB_REG
        db              32h, 64h        ;PWM16_1_PWIDTH_LSB_REG
;     Instance name PWM16_1, Block Name PWM16_MSB(DCA05)
        db              37h, 00h        ;PWM16_1_CONTROL_MSB_REG
        db              35h, 00h        ;PWM16_1_PERIOD_MSB_REG
        db              36h, 00h        ;PWM16_1_PWIDTH_MSB_REG
;  Instance name UART_1, User Module UART
;     Instance name UART_1, Block Name RX(DCA07)
        db              3fh, 00h        ;UART_1_RX_CONTROL_REG
        db              3dh, 00h        ;UART_1_
        db              3eh, 00h        ;UART_1_RX_BUFFER_REG
;     Instance name UART_1, Block Name TX(DCA06)
        db              3bh, 00h        ;UART_1_TX_CONTROL_REG
        db              39h, 00h        ;UART_1_TX_BUFFER_REG
        db              3ah, 00h        ;UART_1_
        db              ffh

; Configuration file trailer Config.asm
```

FIGURE 7C

```
; PSoCConfig.asm
;
; This file is generated by the Device Editor on Application Generation.
; It contains code which loads the configuration data table generated in
; the file ConfigTBL.asm
;

export LoadConfigInit
export_LoadConfigInit
Export LoadConfig_project
export_LoadConfig_project

Flag_CFG_MASK:          equ     10h             ;M8C flag register REG address bit
mask
END_CONFIG_TABLE:       equ     ffh             ;end of config table indicator

_LoadConfigInit:
LoadConfigInit:
                lcall   LoadConfig_project

        ret

;
;Load Configuration project
;
_LoadConfig_project:
LoadConfig_project:
        or              F, FLAG_CFG_MASK                                        ;set for
bank 1
        mov             A, >LoadConfigTBL_project_Bank1  ;load bank 1 table
        mov             X, <LoadConfigTBL_project_Bank1
        call    LoadConfig                                                      ;load the
bank 1 values
        and             F, ~FLAG_CFG_MASK                                       ;switch
to bank 0
        mov             A, >LoadConfigTBL_project_Bank0 ;load bank 0 table
        mov             X, <LoadConfigTBL_project_Bank0
        call    LoadConfig                                                      ;load the
bank 0 values
        ret

;
; LoadConfig
;
; This function is not exported. It assumes that the addresses of the table to be loaded
; is contained in the X and A registers as if a romx instruction is the next instruction
```

FIGURE 8A

```
; to be executed, i.e. lower address in X and upper address in A. There is no return value.
;
LoadConfig:
LoadConfigLp:
        push    X                                       ;save config table address on stack
        push    A
        romx                                            ;load config address
        cmp             A, END_CONFIG_TABLE     ;check for end of table
        jz              EndLoadConfig                   ;if so, end of load
        mov             X, SP                           ;save the address away
        mov             [X], A
        pop             A                                       ;retieve the table address
        pop             X
        inc             X                                       ;advance to the data byte
        jnc             NoOverFlow1                     ;check for overflow
        inc             A                                       ;if so, increment MSB
NoOverFlow1:
        PUSH    x                                       ;save the config table address
again
        push    A
        romx                                            ;load the config data
        mov             X, SP                           ;retrieve the config address
        mov             X, [X]
        mov             reg[X], A               ;write the config data
        pop             A                                       ;retieve the table address
        pop             X
        inc             X                                       ;advance to the next
address
        jnc             NoOverFlow2                     ;check for overflow
        inc             A                                       ;if so, increment MSB
NoOverFlow2:
        jmp             LoadConfigLp                    ;loop back
EndLoadConfig:
        pop             A                                       ;clean up the stack
        pop             A
        ret
```

FIGURE 8B

```
;; ******************************************
;; ******************************************
;; ADCINC12.asm
;;
;; Assembler source for the 12 bit Incremental
;;A/D converter.
;;
;; ******************************************
;;******************************************
;;

export ADCINC12_1_Start
export_ADCINC12_1_Start
export ADCINC12_1_SetPower
export_ADCINC12_1_SetPower
export ADCINC12_1_Stop
export_ADCINC12_1_Stop
export ADCINC12_1_GetSamples
export_ADCINC12_1_GetSamples
export ADCINC12_1_StopAD
export_ADCINC12_1_StopAD
export ADCINC12_1_flsdata
export_ADCINC12_1_flsdata
export ADCINC12_1_GetData
export_ADCINC12_1_GetData
export ADCINC12_1_ClearFlag
export_ADCINC12_1_ClearFlag

include "ADCINC12_1.inc"
include "m8c.inc"

LowByte:    equ 1
HighByte:   equ 0

;; ---------------------------------------------------------------
;; Start:
;; SetPower:
;; Applies power setting to the module's analog
;; Programmable System block
;; INPUTS: A contains the power setting
;; OUTPUTS: None.
;; ---------------------------------------------------------------
ADCINC12_1_Start:
_ADCINC12_1_Start:
ADCINC12_1_SetPower:
_ADCINC12_1_SetPower:
  and A,03h
  or A,f0h
  mov reg[ADCINC12_1_AtoDcr3],A
  ret

;; ---------------------------------------------------------------
;; Stop:
;; SetPower:
;; Removes power setting to the module's analog
;; Programmable System block
;; INPUTS: None
;; OUTPUTS: None.
;; ---------------------------------------------------------------
ADCINC12_1_Stop:
_ADCINC12_1_Stop:
  and reg[ADCINC12_1_AtoDcr3],~03h
  ret

;; ---------------------------------------------------------------
;; Get_Samples:
;; SetPower:
;; Starts the A/D convertor and will place data in
;;memory. A flag
;; is set whenever a new data value is available.
;; INPUTS: A passses the number of samples (0
;;is continuous).
;; OUTPUTS: None.
;; ---------------------------------------------------------------
ADCINC12_1_GetSamples:
_ADCINC12_1_GetSamples:
  mov [ADCINC12_1_blncrC],A       ;number
;of samples
  or reg[INT_MSK1],(ADCINC12_1_TimerMask |
ADCINC12_1_CounterMask)
                          ;enable both interrupts
  Mov[ADCINC12_1_cTimerU],0       ;Force the
;Timer to do one cycle of rest
  or reg{ADCINC12_1_AtoDcr3],10h  ;force the
;Integrator into reset
  mov[ADCINC12_1_cCounterU],ffh   ;Initialize
;Counter

  mov[ADCINC12_1_TimerDR1],ffh
  mov[ADCINC12_1_CounterDR1],ffh
  mov[ADCINC12_1_TimerCR0],01h  ;enable
;the Timer
  mov[ADCINC12_1_flncr],00h       ;A/D Data
;Ready Flag is reset
  ret
```

FIGURE 9A

(Continued)

```
;; ------------------------------------------------------------
;; StopAD:
;; Completely shuts down the A/D in an orderly
;;manner. Both the
;; Timer and Counter Interrupts are disabled.
;; INPUTS: None.
;; OUTPUTS: None.
; ;------------------------------------------------------------
ADCINC12_1_StopAD:
_ADCINC12_1_StopAD:
  mov[ADCINC12_1_TimerCR0],00h
;disable the Timer
  mov[ADCINC12_1_CounterCR0],00h
;disable the Counter
  nop
  nop
  and
reg[INT_MSK1],~(ADCINC12_1_TimerMask |
ADCINC12_1_CounterMask)
                          ;Disable both
;;interrupts
  or reg[ADCINC12_1_AtoDc3],10h            ;reset
;;Intergrator
  ret
;; ------------------------------------------------------------
;; flddata:
;; Returns the status of the A/D Data
;; is set whenever a new data value is available.
;; INPUTS: None.
;; OUTPUTS: A returned data status A=:0 no
;;data available
;;                !=: 0 data available.
; ;------------------------------------------------------------
ADCINC12_1_flsdata:
_ADCINC12_1_flsdata:
  movA,[ADCINC12_1_flncr]
  ret
;; ------------------------------------------------------------
;; iGetData:
;; Returns the status from the A/D. Does not
;;check if data is
;; available.
;; is set whenever a new data value is available.
;; INPUTS: None.
;; OUTPUTS: X:A returns the A/D data value.

; ;------------------------------------------------------------
ADCINC12_1_iGetData:
_ADCINC12_1_iGet Data:
  mov X,[(ADCINC12_1_ilncr+HighByte)]
  mov A,[(ADCINC12_1_ilncr+LowByte)]
  ret

;;------------------------------------------------------------
;; ClearFlag:
;; clears the data ready flag.
;; INPUTS: None
;; OUTPUTS: None.
;; ------------------------------------------------------------
ADCINC12_1_ClearFlag:
_ADCINC12_1_ClearFlag:
  mov [ADCINC12_1_flncr],00h
  ret

ADCINC12_1_API_End:
```

FIGURE 9B

```
HEADER FILES// ********************************************
// *********************************************
//
// ADCINC12_1.h for the 12 bit incremental A/D converter
//
// C declarations for the ADCINC12 User Module
//
//
// *********************************************
//*********************************************

#define ADCINC12_1_OFF    0
#define ADCINC12_1_LOWPOWER      1
#define ADCINC12_1_MEDPOWER      2
#define ADCINC12_1_HIGHPOWER     3

#pragma fastcall ADCINC12_1_Start
#pragma fastcall ADCINC12_1_SetPower
#pragma fastcall ADCINC12_1_GetSamples
#pragma fastcall ADCINC12_1_StopAD
#pragma fastcall ADCINC12_1_Stop

#pragma fastcall ADCINC12_1_flsData
#pragma fastcall ADCINC12_1_iGetData
#pragma fastcall ADCINC12_1_ClearFlag

extern void ADCINC12_1_Start(char power);
extern void ADCINC12_1_SetPower(char power);
extern void ADCINC12_1_GetSamples(char chout);
extern void ADCINC12_1_StopAD(void);
extern void ADCINC12_1_Stop(void);

extern char ADCINC12_1_flsData(void);
extern int ADCINC12_1_iGetData(void);
extern void ADCINC12_1_ClearFlag(void);
```

FIGURE 10

```
;; ********************************************
;;
;;
;; ADCINC12_1.inc for the 12 bit incremental A/D converter
;;
;;
;; Assembler declarations for the ADCINC12 User Module
;;;;
;;;; ********************************************
;; ********************************************
;;

ADCINC12_1_AtoDcr0:         equ     90h
ADCINC12_1_AtoDcr1:         equ     91h
ADCINC12_1_AtoDcr2:         equ     92h
ADCINC12_1_AtoDcr3:         equ     93h
ADCINC12_1_CounterFN:       equ     24h
ADCINC12_1_CounterSL:       equ     25h
ADCINC12_1_CounterOS:       equ     26h
ADCINC12_1_CounterDR0:      equ     24h
ADCINC12_1_CounterDR1:      equ     25h
ADCINC12_1_CounterDR2:      equ     26h
ADCINC12_1_CounterCR0:      equ     27h
ADCINC12_1_TimerFN:         equ     20h
ADCINC12_1_TimerSL:         equ     21h
ADCINC12_1_TimerOS:         equ     22h
ADCINC12_1_TimerDR0:        equ     20h
ADCINC12_1_TimerDR1:        equ     21h
ADCINC12_1_TimerDR2:        equ     22h
ADCINC12_1_TimerCR0:        equ     23h
ADCINC12_1_TimerMask:       equ     01h
ADCINC12_1_CounterMask:     equ     02h
ADCINC12_1_OFF:     equ     0
ADCINC12_1_LOWPOWER:        equ     1
ADCINC12_1_MEDPOWER:        equ     2
ADCINC12_1_HIGHPOWER:       equ     3
ADCINC12_1_NUMBITS:         equ     12
```

FIGURE 11

```
;; ***************************************************
;;
;; ADCINC12int.asm
;;
;; Assembler source for interrupt routines the 12 bit Incremental
;;A/D converter.
;;
;; ***************************************************
;;


export ADCINC12_1_CNT_INT
export_ADCINC12_1_TMR_INT
include "ADCINC12_1.inc"
include "m8c.inc"

Area bss(RAM)
  ADCINC12_1_cTimerU:   BLK 1   ;The upper byte of the Timer
  ADCINC12_1_cCounterU: BLK 1   ;The Upper byte of the Counter
  ADCINC12_1_ilncr:
  ADCINC12_1_ilncr:     BLK 2   ;A/D value
  ADCINC12_1_flncr
  ADCINC12_1_flncr:     BLK 1   ;Data Valid Flag
  ADCINC12_1_blncrC:    BLK 1   ;# of times to run A/D

area text(ROM,REL)

export_ADCINC12_1_cTimerU
export ADCINC12_1_cCounterU
export_ADCINC12_1_ilncr
export ADCINC12_1_ilncr
export_ADCINC12_1_flncr
export ADCINC12_1_flncr
export_ADCINC12_1_blncrC

LowByte:   equ 1
HighByte:   equ 0

;; ---------------------------------------------------------------------------
;; CNT_INT:
;; Decrement the upper (software) half on the counter whenever the
;; lower (hardware) half of the counter underflows.
;; INPUTS:: None.
;; OUTPUTS: None.
;; ---------------------------------------------------------------------------
ADCINC12_1_CNT_INT:
  dec[ADCINC12_1_cCounterU]
  reti
```

FIGURE 12A

```
;; ------------------------------------------------------------------------------
;; TMR_INT:
;; This routine allows the counter to collect data for 64 timer cycles
;; This routine then holds the integrator in reset for one cycle while
;; the A/D value is calculated.
;; INPUTS:: None.
;; OUTPUTS: None.
;; ------------------------------------------------------------------------------
ADCINC12_1_TMR_INT:
  dec[ADCINC12_1_cTimerU]
; if(upper count>=0)
  jc else 1
   reti
  else 1:;(upper count decremented pass 0)
   tst reg{ADCINC_1_AtoDcr3],10h  ;to change when ice is fixed dbz
   jz else2
```

FIGURE 12A (Continued)

```
; if(A/D has been in reset mode)
   mov reg[ADCINC12_1_CounterCR0],01h     ;Enable Counter
   and reg[ADCINC12_1_AtoDcr3],~10h       ;Enable Analog Counter
   mov reg[ADCINC12_1_cTimerU],((1<<(ADCINC12_1_NUMBITS-6))-1)
                                  ;This will be the real counter value
   reti
  else2:;(A/D has been in integrate mode)
   mov reg[ADCINC12_1_CounterCR0],00h     ;disable Counter
   or F,01h                       ;Enable the interrupts
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Good place to add code to switch inputs for multiplexed input to ADC
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
   or reg[ADCINC12_1_AtoDcr3],10h ;Reset Integrator
   mov [(ADCINC12_1_iIncr+LowByte)],ffh
   mov [(ADCINC12_1_iIncr+HighByte)],(ffh-(ADCINC12_1_NUMBITS-7)))
;
   push A
   mov A, reg[ADCINC12_1_CounterDR0],01h ;read Counter
   mov A, reg[ADCINC12_1_CounterDR2],01h ;now you really read the data
   sub[(ADCINC12_1_iIncr+LowByte)],A
   mov A, reg[ADCINC12_1_cCounterU]
   sbb[(ADCINC12_1_iIncr+HighByte)],A
   pop A
   cmp[(ADCINC12_1_iIncr+HighByte)],(1<<(ADCINC12_1_NUMBITS-7))
   jnz endif10
;   if(max positive value)
     dec[(ADCINC12_1_iIncr+HighByte)]
     mov[(ADCINC12_1_iIncr+LowByte)],ffh
   endif10:
   asr[(ADCINC12_1_iIncr+HighByte)]              ;divide by 4
   rrc[(ADCINC12_1_iIncr+LowByte)]
   asr[(ADCINC12_1_iIncr+HighByte)]
   rrc[(ADCINC12_1_iIncr+LowByte)]

   mov[ADCINC12_1_fIncr],01h                     ;set AD data flag
```

FIGURE 12B

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; User code here for interrupt system.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
   cmp[ADCINC12_1_bIncrC],00h
  jz endif3
; if(ADCINC12_1_bIncrC is not zero)
      dec[ADCINC12_1_bIncrC]
      jnz endif4
; if(ADCINC12_1_bIncrC has decremented down to zero to 0))
      mov reg[ADCINC12_1_TimerCR0],00h            ;disable the Timer
      mov reg[ADCINC12_1_CounterCR0],00h          ;disable the Counter
      nop
      nop
      and reg[INT_MSK1],~(ADCINC12_1_TimerMask | ADCINC12_1_CounterMask)
                                          ;disable both interrupts
      or reg[ADCINC12_1_AtoDcr3],10h                      ;reset Integrator
      reti
    endif4:;
   endif3:;
  endif2:;
  mov [ADCINC12_1_cTimerU],00h          ;Set Timer for one cycle of reset
  mov [ADCINC12_1_cCounterU],ffh        ;Set Counter hardware for easy enable
  mov reg[ADCINC12_1_CounterDR1],ffh
  reti
 endif1:;

ADCINC12_1_APIINT_End: A/D converter
```

FIGURE 12B (Continued)

1300

```
;-----------------------------------------------------------
; Interrupt Vector Table
;-----------------------------------------------------------
;
; Interrupt vector table entries are 4 bytes long
; and contain the code
; that services the interrupt (or causes it to be
; serviced).
;
;-----------------------------------------------------------

  Area    TOP(ROM,ABS)

  org 0                    ;Reset Interrupr Vector
  jmp_start                   ;First Instruction
; executed following a Reset

  org 04h                  ;Supply Monitor Interrupt
; Vector
  //call void_handler
  reti

  org08h          Block DBA00
; Interrupt Vector
  ljmp  ADCINC12_1_TMR_INT
  reti

  org0Ch          ;Block DBA01
; Interrupt Vector
  ljmp  ADCINC12_1_CNT_INT
  reti

  org10h          ;Block DBA02
; Interrupt Vector
  //call void_handler
  reti

  org14h          ;Block DBA03
; Interrupt Vector
  ljmp  Counter16_1INT
  reti

   org 18h       ;Block DCA04
; Interrupt Vector
//call void_handler
  reti

   org 1Ch              ;Block DCA0
; Interrupt Vector
  ljmp  PWM16_1INT
  reti

   org 20h              ;Block DCA06
; Interrupt Vector
  ljmp  UART_1TX_INT
  reti

   org 24h              ;Block DCA07
; Interrupt Vector
  ljmp  UART_1RX_INT
  reti

   org 28h              ;Analog Column 0
; Interrupt Vector
  //call void_handler
  reti

   org 2Ch              ;Analog Column 1
; Interrupt Vector
  //call void_handler
  reti

   org 30h              ;Analog Column 2
; Interrupt Vector
  //call void_handler
  reti

   org 34h              ;Analog Column 3
; Interrupt Vector
  //call void_handler
  reti

   org 38h              ;GPIO Interrupt Vector
  //call void_handler
  reti

   org 3Ch              ;Sleep Timer Interrupt
; Vector
  jmp  SleepTimerISR
  reti
```

FIGURE 13A

1351 boot.asm Interrupt Name	1352 Data Sheet Interrupt Name	1353 Type
Start	Reset	Fixed
Interrupt1	Supply Monitor	Fixed
Interrupt2	DBA00	Programmable System Block
Interrupt3	DBA01	Programmable System Block
Interrupt4	DBA02	Programmable System Block
Interrupt5	DBA03	Programmable System Block
Interrupt6	DCA04	Programmable System Block
Interrupt7	DCA05	Programmable System Block
Interrupt8	DCA06	Programmable System Block
Interrupt9	DCA07	Programmable System Block
Interrupt10	Analog Column 0	Programmable System Block
Interrupt11	Analog Column 1	Programmable System Block
Interrupt12	Analog Column 2	Programmable System Block
Interrupt13	Analog Column 3	Programmable System Block
Interrupt14	GPIO	Fixed
Interrupt15	SLEEP TIMER	Fixed

FIGURE 13B

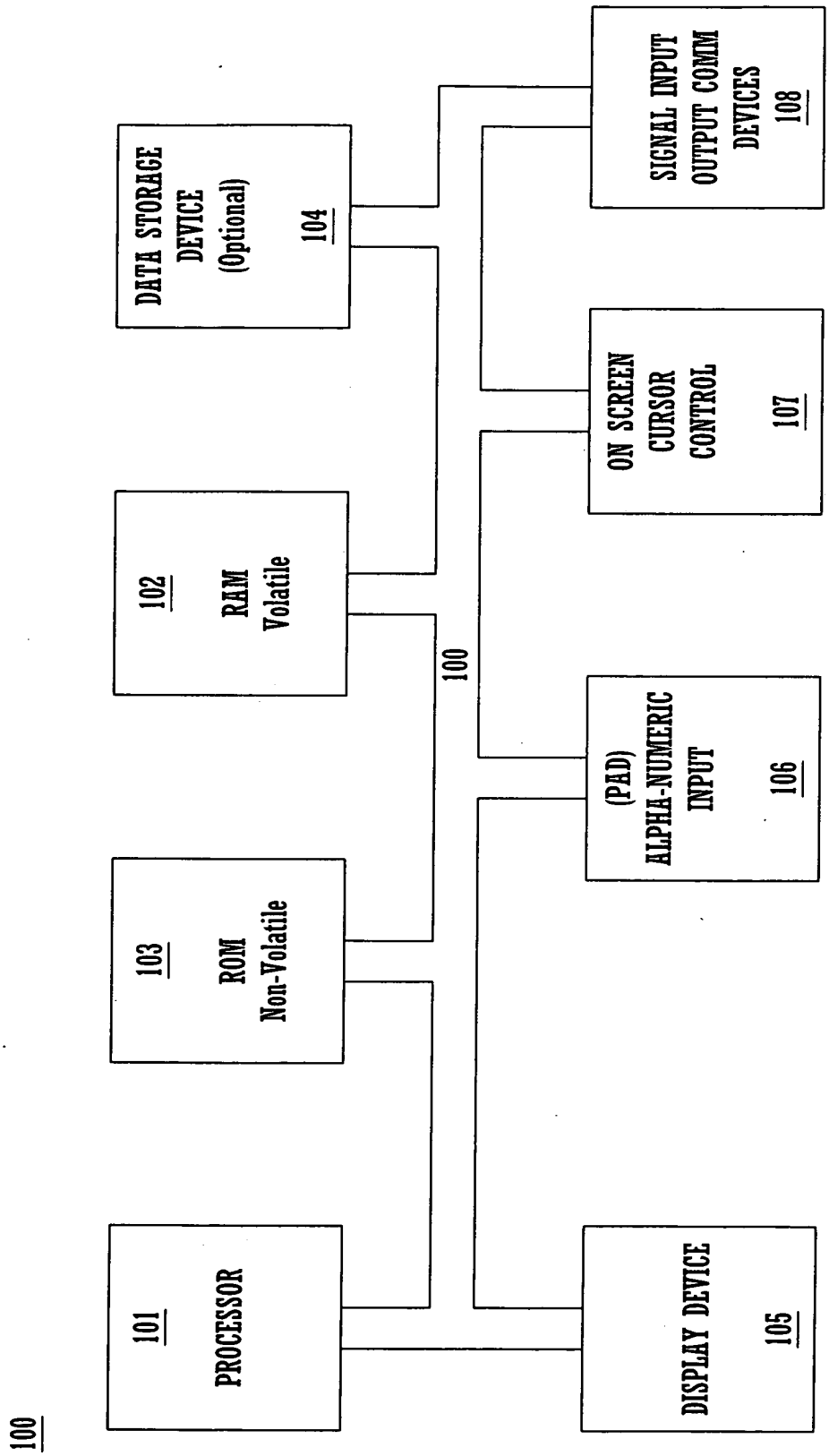

100
101
PROCESSOR
103
ROM
Non-Volatile
102
RAM
Volatile
DATA STORAGE
DEVICE
(Optional)
104
100
DISPLAY DEVICE
105
(PAD)
ALPHA-NUMERIC
INPUT
106
ON SCREEN
CURSOR
CONTROL
107
SIGNAL INPUT
OUTPUT COMM
DEVICES
108

FIGURE 14